1
1 ej

Universidad La Salle

ESCUELA DE INGENIERIA
Incorporada a la U.N.A.M.

INTRODUCCION AL CONCEPTO DE INTELIGENCIA ARTIFICIAL MEDIANTE LA IMITACION DE UN PROCESO DE APRENDIZAJE

TESIS PROFESIONAL

Que para obtener el Titulo de
INGENIERO MECANICO - ELECTRICO
AREA PRINCIPAL EN SISTEMAS ELECTRICOS, ELECTRONICOS Y DE COMUNICACION

p r e s e n t a

JUAN JOSE TALAMANTES DEL TORO

TESIS CON
FALLA DE ORIGEN

MEXICO, D. F. 1983

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

INTRODUCCION AL CONCEPTO DE INTELIGENCIA ARTIFICAL MEDIANTE LA IMITACION DE UN PROCESO DE APRENDIZAJE.
(SOLUCION A UN LABERINTO)

I N D I C E

1.- OBJETIVO DE LA TESIS

CON ESTE TRABAJO SE PRETENTE DAR UNA IDEA DE LOS PRINCIPIOS Y EL SIGNIFICADO DEL CONCEPTO "INTELIGENCIA ARTIFICIAL", CON PALABRAS SENCILLAS Y DIRIGIDAS AL COMUN DE LA GENTE, QUE TENIENDO O NO CONOCIMIENTOS DE INGENIERIA O DE COMPUTACION, SERA CAPAZ DE ENTENDERLO, CRITICARLO Y MEJORARLO SEGUN LA VISION QUE TENGA DEL MUNDO QUE LO RODEA Y DE SI MISMO.

AUNQUE EL TITULO PUEDA PARECER DEMASIADO PRETENSIOSO, NO ES OTRA COSA QUE EL RESUMEN DEL PROPOSITO DE LA TESIS, Y CREO QUE DA UNA IDEA CLARA DEL ALCANCE DE LA MISMA.

EL CUERPO DE ESTE ESCRITO ESTA FORMADO POR TRES PARTES PRINCIPALES:

* CONCEPTOS TEORICOS
** ANALISIS ESPECIFICO DE UN PROBLEMA
*** SIMULACION DE LA SOLUCION

EL DESARROLLO DE LA TESIS VA DE LO GENERAL A LO PARTICULAR, PUES EL PRIMER PUNTO SE CENTRA EN ESCLARECER EL SIGNIFICADO DEL CONCEPTO, Y UNA VEZ TERMINADA ESTA PARTE, NOS ENFOCAMOS A LA SOLUCION DE UN CASO EN ESPECIAL, EL CUAL SE JUSTIFICA A SU DEBIDO TIEMPO, Y QUE EJEMPLIFICA, DE ALGUNA MANERA, LOS CONCEPTOS TEORICOS EXPUESTOS.

ASI PUES, AL COMENZAR EL ANALISIS DEL CASO, ES CUANDO EL LECTOR PODRA EMPEZAR A CRITICAR EL TRATAMIENTO DEL MISMO, APROBANDOLO O REPROBANDOLO SEGUN EL CASO, CUMPLIENDO ASI, CON OBJETIVO PRIMORDIAL DE ESTA TESIS.

EL TERCER PUNTO NO ES OTRA COSA QUE LA PLANEACION, DESARROLLO E IMPLEMENTACION DE UN SISTEMA O PROGRAMA CAPAZ DE SIMULAR LA SOLUCION ENCONTRADA DURANTE EL ANALISIS. SE DIJO ANTES QUE LA TESIS ESTABA DIRIGIDA A EL COMUN DE LA GENTE, PERO ESTE PASO PUEDE QUE NO SEA SENCILLO DE ENTENDER, PUES HAY QUE TENER EN CUENTA QUE EN LA IMPLEMENTACION DE UN SISTEMA O PROGRAMA EN UNA COMPUTADORA, SE DEBEN MANEJAR CONCEPTOS PROPIOS DE LA MAQUINA, A LOS CUALES NO TODO MUNDO TIENE ACCESO.
SIRVA PUES ESTE TERCER PASO SOLO COMO COMPROBACION DE QUE LA SOLUCION ENCONTRADA DURANTE EL ANALISIS, REALMENTE FUNCIONA.

RESUMIENDO, PODEMOS AFIRMAR QUE LA META DE LA TESIS, ES LANZAR UN RETO Y UNA INVITACION A LA GENTE, PARA QUE EMPIEZE A INTERESARSE EN LA INVESTIGACION DE ESTA RAMA DE LA CIENCIA, QUE PROMETE MUCHAS SOLUCIONES A PROBLEMAS QUE ANTES SE CREIAN, QUE SOLO EL HOMBRE PODRIA RESOLVER.

2.- QUE SE ENTIENDE POR INTELIGENCIA ARTIFICIAL ?

ESTE ES UN CONCEPTO UN TANTO ABSTRACTO, PUES DE HECHO SE TENDRIA QUE EMPEZAR POR DEFINIR QUE ES EN REALIDAD LA INTELIGENCIA, YA QUE ESTA ES ALGO MUCHO MAS COMPLEJO QUE EL NUMERO QUE SE OBTIENE MEDIANTE LAS PRUEBAS DE COCIENTE INTELECTUAL. ADEMAS PODRIAMOS CUESTIONAR HASTA QUE PUNTO ES VALIDO EL USO DEL TERMINO 'INTELIGENCIA ARTIFICIAL' O MAS COMUNMENTE LLAMADA 'AI'(POR SUS SIGLAS EN INGLES). COMO ESTE NO ES UN TRATADO DE FILOSOFIA, NO NOS INTERESA TANTO LA RAZON POR LA CUAL SE ESCOGIO ESTE TERMINO, SI NO CUAL ES SU SIGNIFICADO, Y QUE AREAS DE LA CIENCIA DE LA COMPUTACION REQUIEREN DE ESTA DISCIPLINA.

PODEMOS DEFINIR LA INTELIGENCIA ARTIFICIAL COMO LA CIENCIA QUE SE DEDICA A LA CREACION DE MAQUINAS QUE SEAN CAPACES DE REALIZAR COSAS, QUE SE DIRIA QUE REQUIEREN INTELIGENCIA SI FUESEN REALIZADAS POR EL HOMBRE.

EN EL AREA DE LA COMPUTACION SE HAN TOMADO DOS DIRECCIONES DIFERENTES EN CUANTO A LA FORMA DE ATACAR EL PROBLEMA DE LA INTELIGENCIA ARTIFICIAL. UNO DE LOS ENFOQUES TRATA DE QUE LAS MAQUINAS HAGAN TRABAJOS INTERESANTES Y COMPLEJOS SIN IMPORTAR EL MODO EN QUE EL HOMBRE LOS REALIZA. EL OTRO TRATA DE SIMULAR EL COMPORTAMIENTO COGNOSCITIVO HUMANO, Y SE LE CONOCE COMO 'MODELACION PSICOLOGICA MEDIANTE COMPUTADORA'.

DE HECHO, EL QUE UNA MAQUINA LLEGUE A RESOLVER PROBLEMAS A UN NIVEL COMPARABLE AL SER HUMANO, ES UNA META SUMAMENTE LEJANA. AUN ASI, YA EXISTEN MUCHAS APLICACIONES INDUSTRIALES DE ROBOTS, QUE DE UNA FORMA U OTRA TIENEN QUE APRENDER LAS TAREAS QUE SE LE ASIGNAN.

PODEMOS ENGLOBAR DENTRO DEL CONCEPTO DE 'AI', CUALQUIER PROCESO QUE REQUIERA DE EL APRENDIZAJE, ES DECIR, QUE LA MAQUINA EN CUESTION SEA CAPAZ DE PEDIRNOS LA INFORMACION NECESARIA, Y QUE APRENDA ESTA INFORMACION PARA PODER USARLA EN EL FUTURO EN UN CASO SIMILAR.

DENTRO DE LOS CAMPOS QUE LA 'AI' EMPEZO A INVADIR MAS RAPIDAMENTE, ESTA EL DE LOS JUEGOS, PUES ESTOS BRINDAN LA OPORTUNIDAD DE CREAR PROGRAMAS, YA QUE LA ACTIVIDAD DE LOS MISMOS ESTA PERFECTAMENTE DEFINIDA. COMO EJEMPLO TENEMOS EL AJEDREZ, EL CUAL A SIDO JUGADO POR SIGLOS, Y EL HOMBRE SIEMPRE HA QUERIDO CREAR UNA MAQUINA QUE LO JUEGUE. EXISTIERON VARIOS INTENTOS, ENTRE ELLOS EL DEL INVENTOR HISPANO L. TORRES Y QUEVEDO, EL CUAL CREO UN DISPOSITIVO FISICO CAPAZ DE REALIZAR EL MATE DE REY Y TORRE CONTRA REY. ESTO LO HIZO EN 1914, Y AUNQUE EL ALGORITMO NO ES DIFICIL, SU REALIZACION FISICA FUE UNA PROEZA.

A LA FECHA SE HAN CREADO MUCHAS MAQUINAS QUE JUEGAN AJEDREZ A UN NIVEL DE AMATEUR. LOS CIENTIFICOS EN ESTA RAMA, SE DIVIDEN EN LOS DOS GRUPOS PRIMARIOS, A SABER: LOS QUE DESEAN IMITAR LA FORMA EN QUE EL HOMBRE JUEGA AJEDREZ, Y AQUELLOS QUE ATACAN EL PROBLEMA DE UNA FORMA MAS DIRECTA. SIN EMBARGO TODAVIA SE ESTA LEJOS DE LOGRAR UNA MAQUINA QUE SEA CAPAZ DE JUGAR EL AJEDREZ COMO UN GRAN MAESTRO.

TAMBIEN SE HA ANALIZADO EL JUEGO DE DAMAS, EL CUAL DESDE LUEGO ES MUCHO MAS SENCILLO QUE EL AJEDREZ, Y SE HA LOGRADO UN ALGORITMO MUY BUENO.

DE HECHO, SE A LOGRADO LA AUTOMATIZACION DE CASI TODOS LOS JUEGOS

CREADOS POR EL HOMBRE, Y ASI MISMO A RAIZ DE LAS INVESTIGACIONES SE HAN CREADO MUCHOS OTROS JUEGOS SUMAMENTE INTERESANTES.

DESDE LUEGO, EL ANALISIS DE CIERTOS PROBLEMAS, LLEVO A LA CONCLUSION DE QUE LOS LENGUAJES DE PROGRAMACION EXISTENTES NO SIEMPRE SERIAN ADECUADOS PARA EL MANEJO DE LA 'AI'. POR LO CUAL SE HAN CREADO VARIOS LENGUAJES ESPECIALES PARA ESTE PROPOSITO, ENTRE ELLOS EL 'LISP', EL CUAL MANEJA LA INFORMACION EN LISTAS CUYA LONGITUD PUEDE VARIAR DURANTE LA COMPUTACION, Y A SU VEZ ESTAS LISTAS PUEDEN CONTENER SUBLISTAS ASOCIADAS A LOS ELEMENTOS DE LA LISTA PRINCIPAL. OTRO LENGUAJE CREADO ES EL DE PROCESAMIENTO DE DATOS EN HILERA, EL CUAL TAMBIEN PROCESA LOS DATOS EN LARGAS LISTAS, LAS CUALES NO TIENEN SUBLISTAS, Y CUYO NOMBRE ES: 'SNOBOL'. ADEMAS MUCHAS PERSONAS HAN UTILIZADO 'FORTRAN' O INCLUSO ENSAMBLADOR. EN ESTE CASO EN PARTICULAR USAREMOS EL BASIC, POR SER EL MAS CONOCIDO, Y POR CONTAR CON UNA MAQUINA CUYO BASIC ES RELATIVAMENTE PODEROSO.

UNA DE LAS CUESTIONES MAS IMPORTANTES ES LA PLANIFICACION. EN ESTA SE TRATARA DE GUIAR A LA MAQUINA PARA QUE BUSQUE LA INFORMACION BASANDOSE EN CRITERIOS OBTENIDOS DE UNA PERCEPCION GLOBAL DEL PROBLEMA. DE ESTA MANERA SE DISMINUYE EL NUMERO DE OPERACIONES QUE TIENE QUE REALIZAR LA COMPUTADORA.

ADEMAS SE PUEDE CONTAR CON UNA FUNCION DE PROXIMIDAD, ES DECIR DE ALGUNA MANERA LA MAQUINA DEBE SABER QUE YA ESTA CERCA DE LA RESOLUCION DE UN PROBLEMA. ESTO DEPENDE DESDE LUEGO DEL TIPO DE APLICACION.

POR OTRO LADO SE LE DEBE DE DAR A LA MAQUINA ALGUNA MANERA DE ADAPTARSE A LOS POSIBLES CAMBIOS QUE PUEDAN OCURRIR EN CUANTO A LA INFORMACION QUE ELLA RECIBE PARA SER PROCESADA. ES DECIR QUE SU APRENDIZAJE PUEDE NO SOLO SER IMPORTANTE EN LA RESOLCION DE UN PROBLEMA, SINO EN LA FORMA DE OBTENER LOS DATOS DEL MISMO. COMO EJEMPLO PODEMOS PONER UNA MAQUINA CAPAZ DE RECONOCER MENSAJES CIFRADOS EN CODIGO MORSE, QUE HAYAN SIDO ESCRITOS A MANO. SI EL OPERADOR NO ES MUY CUIDADOSO, PUEDE EMPEZAR A HACER LOS PUNTOS Y LAS RAYAS DE UNA LONGITUD DETERMINADA. PERO DESPUES DE UN TIEMPO O AL DIA SIGUIENTE PUEDE QUE ESTA LONGITUD SEA VARIADA, Y ENTONCES SI LA MAQUINA NO TIENE EL CRITERIO PARA PODER RECONOCER ESTA SITUACION, INTERPRETARA EL MENSAJE DE DIFERENTE MANERA. EN ESTE PUNTO ENCONTRAMOS LO QUE SE PUEDE DENOMINAR COMO RECONOCIMIENTO DE FORMAS, Y QUE ES UN PROCESO ABSTRACTO QUE PUEDE SER INCLUIDO EN CUALQUIER PROGRAMA DE INTELIGENCIA ARTIFICIAL.

SE HAN LOGRADO MUCHOS AVANCES IMPORTANTES EN ESTA RAMA DE LA CIENCIA Y EN OTRAS RAMAS RELACIONADAS CON ELLA. POR EJEMPLO TENEMOS LAS INVESTIGACIONES QUE SE ESTAN HACIENDO EN EL CAMPO DEL RECONOCIMIENTO DE PALABRA, ES DECIR, QUE UNA COMPUTADORA O MAQUINA SEA CAPAZ DE RECIBIR ORDENES HABLADAS POR CUALQUIER PERSONA. ESTA CAPACIDAD ES SUMAMENTE IMPORTANTE, YA QUE EXISTEN CIERTAS ACTIVIDADES HUMANAS QUE REQUIEREN DEL USO DE UNA COMPUTADORA, LA CUAL ES PROGRAMADA EN UN PRINCIPIO, Y DESPUES TIENE QUE RECIBIR UN NUMERO LIMITADO DE COMANDOS PARA PODER FUNCIONAR LOS CUALES PODRIAN DARSELE EN FORMA ORAL, AHORRANDO EL USO TECLADOS QUE EN MUCHOS CASOS PUEDEN SER DIFICILES DE USAR. ADEMAS EXISTEN DOS METAS SEPARADAS EN ESTE CAMPO: LA PRIMERA, QUE YA HA SIDO LOGRADA, ES EL RECONOCIMIENTO DE PALABRAS O FRASES AISLADAS, LO CUAL NO TIENE MUCHO QUE VER

CON LA INTELIGENCIA ARTIFICIAL. LA OTRA META, QUE TODAVIA NO SE HA LOGRADO ES LA DE CREAR UNA MAQUINA QUE SEA CAPAZ DE ENTENDER PERFECTAMENTE EL SIGNIFICADO DE UNA CONVERSACION NORMAL, EL CUAL PUEDE VARIAR DE MUCHAS MANERAS, POR FACTORES TALES COMO ENTONACION, ORDEN DE LAS PALABRAS, CONTEXTO PROPIO DE LAS ORACIONES ETC. ASI PUES, EL ENTENDIMIENTO DEL LENGUAJE ES UNA AREA MUY GRANDE DE INVESTIGACION.

EN EL AREA DEL HABLA, SE HAN LOGRADO MICROSISTEMAS, QUE SON CAPACES DE HABLAR DE MANERA QUE RESULTA IMPOSIBLE DISTINGUIR SI LO ESCUCHADO PROVIENE DE UNA COMPUTADORA, O BIEN DE UNA GRABACION ANALOGICA. ASI PUES SE TIENEN JUEGOS QUE HABLAN; COMPUTADORAS PERSONALES, CAPACES DE TRANSFORMAR LO QUE SE ESTA ESCRIBIENDO EN UN TECLADO, A UN VOZ HUMANA PERFECTAMENTE RECONOCIBLE. ESTO ULTIMO EN PARTICULAR ES SUMAMANTE UTIL PARA OPERARIOS CIEGOS, QUE DE ALGUNA MANERA PUEDEN CORROBORAR LO QUE ESTAN ESCRIBIENDO EN EL TECLADO. ALGUNAS COMPUTADORAS PERSONALES, ACEPTAN YA COMANDOS HABLADOS, LOS CUALES PUEDEN SER DEFINIDOS POR EL MISMO USUARIO, SIN LIMITES, MAS QUE EN LA CAPACIDAD PROPIA DE LA MAQUINA.

LA VISION, ES OTRO DE LOS PROBLEMAS QUE SE HA ATACADO, Y SE HA LOGRADO TENER ROBOTS CAPACES DE TOMAR PIEZAS ORIENTADAS AL AZAR, DE UNA SUPERFICIE. ASI MISMO, SON CAPACES DE RECONOCER PIEZAS QUE YA HAYAN VISTO CON ANTERIORIDAD.

UNO DE LOS PROBLEMAS PRINCIPALES CON LOS QUE SE HAN TOPADO TODAS ESTAS INVESTIGACIONES, ES QUE TODAVIA NO EXISTEN MAQUINAS CAPACES DE SOPORTAR TODOS LOS CONCEPTOS QUE SE TIENEN QUE MANEJAR EN LA RESOLUCION DE CIERTOS PROBLEMAS. COMO EJEMPLO PODEMOS CITAR EL DE LA VISION, EL CUAL EN CIERTA FORMA TIENE QUE MANEJAR UNA GRAN CANTIDAD DE INFORMACION, Y POR LO TANTO SE NECESITAN MAQUINAS CADA VEZ MAS PODEROSAS, QUE PUEDAN INTEPRETARLA TAN RAPIDO COMO EL CEREBRO HUMANO.

PARA LOGRAR ESTAS MAQUINAS SE TIENE QUE AVANZAR UN PASO MAS EN EL MUNDO DE LA MICROELECTRONICA, ES DECIR CREAR CIRCUITOS CON DENSIDADES CIENTOS DE VECES MAYORES QUE LAS EXISTENTES, CON TECNICAS TOTALMENTE DIFERENTES A LAS QUE SE UTILIZAN HOY EN DIA, Y A PRECIOS TODAVIA MAS BAJOS, DE MANERA QUE LAS INVESTIGACIONES EN ESTE CAMPO NO SE QUEDAN EN MEROS CONCEPTOS TEORICOS, FACTIBLES DE SER MANEJADOS SOLAMENTE POR LAS SUPERCOMPUTADORAS EXISTENTES, SI NO QUE LLEGEN A SER INTEGRADOS A LAS LABORES COTIDIANAS DEL HOMBRE.

3.- QUE SIGNIFICA IMITACION DE UN PROCESO DE APRENDIZAJE ?

EL APRENDIZAJE ES UN MECANISMO QUE SE PRESENTA EN CASI TODAS LAS ACTIVIDADES DEL SER HUMANO, PUES AUNQUE ESTAS SEAN ACTIVIDADES TOTALMENTE MECANICAS, LA MENTE HUMANA ESTA CONSTANTEMENTE CAPTANDO NUEVAS IMPRESIONES QUE A FINAL DE CUENTAS SE ENGLOBAN EN UN APRENDIZAJE TOTAL.

AUNQUE ES DIFICIL DEFINIR QUE MECANISMOS USAN O IMITAN UN PROCESO DE APRENDIZAJE, PODEMOS DECIR QUE EN EL AREA DE LA INTELIGENCIA ARTIFICIAL SE ENTIENDE POR APRENDIZAJE, CUALQUIER PROCESO EN EL CUAL UNA MAQUINA ADQUIERA INFORMACION SIGNIFICATIVA DEL EXTERIOR, LA GUARDE ORDENADAMENTE, Y SEA CAPAZ DE UTILIZARLA PARA FACILITARSE A SI MISMA LAS TAREAS QUE ESTE REALIZANDO. EN OTRAS PALABRAS, SE NECESITA QUE LA MAQUINA NO SOLAMENTE GUARDE LA INFORMACION, SINO QUE ESTA MODIFIQUE EL COMPORTAMIENTO DE COMPUTADORA DE ALGUNA MANERA.

LAS FORMAS DE APRENDIZAJE INCLUYEN:

- MEMORIZACION DIRECTA DE CASOS PREVIOS SIN GENERALIZACION A CASOS SIMILARES; ESTO ES, 'APRENDIZAJE MAQUINAL'.

- LA VARIACION DEL MECANISMO DE ACUERDO A LOS CAMBIOS SUFRIDOS POR LOS PARAMETROS NUMERICOS, LOS CUALES SE MODIFICAN CON EL OBJETO DE OPTIMIZAR EL COMPORTAMIENTO.

- GENERALIZAR LA SOLUCION DE SITUACIONES SIMILARES. ESTE SE REALIZA MEDIANTE INDUCCION, Y SE REQUIERE UNA ESTRUCTURA DE DATOS MUY COMPLETA Y UN MANEJO DE LOS MISMOS DE UNA MANERA ESPECIAL.

- REESCRITURA DE PARTES DEL PROGRAMA AUTOMATICAMENTE.

PARA QUE LA MAQUINA PUEDA APRENDER, DEBE ENTERARSE DE ALGUNA MANERA QUE SU COMPORTAMIENTO ESTA MEJORANDO, O QUE YA TUVO EXITO, Y A ESTO ES A LO QUE SE LLAMA 'REFUERZO'. ADEMAS LA COMPUTADORA DEBE TENER UNA CAPACIDAD DE MEMORIA SUFICIENTEMENTE GRANDE PARA MANEJAR CON FACILIDAD TODA LA INFORMACION GENERADA DURANTE EL PROCESO.

A FINAL DE CUENTAS MUCHOS INVESTIGADORES HAN LLEGADO A LA CONCLUSION DE QUE EL METODO IDEAL PARA LA RESOLUCION DE PROBLEMAS COMPLEJOS EN EL FUTURO, SERA EL SISTEMA HOMBRE-MAQUINA, ES DECIR SISTEMAS INTERACTIVOS EN LOS CUALES EL HOMBRE REALIZARA LAS COSAS QUE HACE MEJOR, PRINCIPALMENTE EL RECONOCIMIENTO DE FORMAS Y LAS INTUICIONES SUTILES NO FORMULADAS EN UN PROGRAMA , MIENTRAS QUE LA COMPUTADORA USARA SUS CUALIDADES DE MEMORIA INFALIBLE, BUSQUEDA RAPIDA Y PRECISION NUMERICA, QUE COMPLEMENTAN LAS HABILIDADES HUMANAS. EL HOMBRE FORMULARA EL PROBLEMA Y SUGERIRA METODOS. EN PUNTOS DE DIFICIL DECISION LA COMPUTADORA RECURRIRA AL HOMBRE PARA PEDIR CONSEJO.

POR ULTIMO REPETIREMOS QUE EL FACTOR MAS IMPORTANTE EN LA CREACION DE UN SISTEMA 'INTELIGENTE', ES LA MANERA EN QUE LA INFORMACION SEA MANEJADA, PUES DE LOS NEXOS O CONEXIONES QUE SE HAGAN ENTRE LOS DATOS, ASI

COMO LA ADECUADA CLASIFICACION DE LA INFORMACION OBTENIDOS CADA VEZ QUE LA MAQUINA REPITA EL PROCESO, DEPENDERA EL QUE EL SISTEMA SEA CAPAZ DE MEJORAR SU COMPORTAMIENTO FRENTE A PROBLEMAS SIMILARES.

4.- PLANTEAMIENTO

EL SISTEMA QUE SE TRATARA DE HACER, ES UN SIMULADOR QUE IMITARA EL COMPORTAMIENTO DE UN ENTE, QUE ESTARA COLOCADO EN CUALQUIER PARTE DE UN LABERINTO, Y USANDO UN RAZONAMIENTO LOGICO, TRATARA DE SALIR DEL MISMO.

UNA VEZ QUE EL SIMULADOR HAYA PODIDO SALIR DEL LABERINTO, RECORDARA LA RUTA QUE TUVO QUE SEGUIR PARA PODER SALIR, Y SERA CAPAZ DESDE LUEGO DE REGRESAR AL PUNTO DE PARTIDA, O BIEN VOLVER A SALIR DEL LABERINTO CADA VEZ QUE SEA COLOCADO EN EL PUNTO DE DONDE PARTIO.

CON ESTO BASTARIA PARA PODER AFIRMAR QUE SE ESTA REALIZANDO UN PROCESO DE INTELIGENCIA ARTIFICIAL, PUES EL MECANISMO CUMPLE POR LO MENOS CON LA CLASIFICACION MAS BAJA DE LOS RANGOS DE APRENDIZAJE VISTOS EN EL CAPITULO ANTERIOR. ES DECIR, QUE LA MAQUINA TIENE UN 'APRENDIZAJE MAQUINAL'.

DEBIDO AL PROBLEMA QUE SE ESTA PLANTEANDO, SABEMOS QUE EL CONOCIMIENTO QUE EL MECANISMO OBTENGA DE LA RESOLUCION DE UN LABERINTO, NO LE PODRA SERVIR DE NADA EN EL MOMENTO QUE TENGA QUE RESOLVER OTRO, PUESTO QUE CADA LABERINTO ES, O PUEDE SER, TOTALMENTE DIFERENTE DE LOS DEMAS, LO QUE HACE IMPOSIBLE ASOCIARLOS. EN OTRAS PALABRAS, NO PODEMOS GENERALIZAR UNA SOLUCION PARA TODOS LOS LABERINTOS.

POR OTRA PARTE, SABEMOS QUE EL ALGORITMO QUE SE USARA EN LA RESOLUCION, ESTARA TOTALMENTE DEFINIDO, LO QUE IMPLICARA QUE NO HABRA VARIACION ALGUNA EN LOS PARAMETROS, QUE PUDIERA LLEVAR A UNA OPTIMIZACION DEL COMPORTAMIENTO. POR LA MISMA RAZON RESULTA INUTIL TRATAR DE UTILIZAR EL NIVEL MAS AVANZADO DE LAS FORMAS DE APRENDIZAJE, QUE ES LA REESCRITURA AUTOMATICA DE PARTES EL PROGRAMA, QUE POR OTRA PARTE SERIA IMPOSIBLE DADO QUE EL LENGUAJE QUE SE VA A UTILIZAR ES EL BASIC, Y NO PERMITE REALIZAR ESTE TIPO DE MANEJOS.

SIN EMBARGO, LA MAQUINA NO ES CAPAZ DE SABER QUE SE ENCUENTRA EN UN LABERINTO DIFERENTE CADA VEZ, POR LO QUE TRATARA DE UTILIZAR EL CONOCIMIENTO ADQUIRIDO ANTERIORMENTE, CADA VEZ QUE TENGA QUE RESOLVER OTRO LABERINTO. ES COMO UN SER HUMANO QUE EN UN MOMENTO DADO ES PUESTO EN UNA CIUDAD QUE DESCONOCE DEL TODO, Y QUE PRETENDE LLEGAR AL CENTRO DE LA MISMA SIN PREGUNTAR A NADIE. LO QUE HARA SERA ENCONTRAR EL CAMINO POR EL METODO DE PRUEBA Y ERROR, Y UNA VEZ QUE LO HALLE, PODRA RECORDAR EL CAMINO QUE SIGUIO PARA LLEGAR A ESE LUGAR. SUPONGAMOS QUE AL DIA SIGUIENTE ESTA PERSONA DESPIERTA, Y SE ENCUENTRA CON QUE NO ESTA EN EL CENTRO, NI EN EL LUGAR DESDE EL CUAL PARTIO EL DIA ANTERIOR. SI LA PERSONA NO TIENE FORMA DE COMUNICARSE NORMALMENTE CON LA GENTE,NO PUEDE SABER NI SIQUIERA SI SE ENCUENTRA EN LA MISMA CIUDAD. PERO NO TIENE PORQUE PENSAR QUE ESTA EN OTRA, POR LO CUAL TRATARA DE LLEGAR AL CENTRO, DE LA MISMA FORMA QUE EL DIA ANTERIOR, SOLO QUE ESTARA FIJANDOSE EN TODO MOMENTO, PARA TRATAR DE ENCONTRAR ALGUN LUGAR CONOCIDO DE ANTEMANO, PARA PODER LLEGAR MAS DIRECTAMENTE. SI CADA DIA QUE PASA HACEMOS QUE LA PERSONA AMANEZCA EN UN LUGAR DIFERENTE, SIEMPRE TENDRA QUE REPETIR EL PROCESO, PERO TENDRA A SU VEZ MAS CONOCIMIENTO DE LA CIUDAD, LO CUAL IMPLICA QUE CADA VEZ PODRA, LLEGAR MAS RAPIDAMENTE AL CENTRO.

A LO QUE SE TRATA DE LLEGAR, ES AL HECHO DE QUE EL SIMULADOR QUE SE HARA, DEBERA SER CAPAZ DE UTILIZAR LA INFORMACION ADQUIRIDA CON ANTERIORIDAD CADA VEZ QUE PRETENDA SALIR DE UN LABERINTO. Y DE ESTA MANERA, SI EL SIMULADOR ES COLOCADO EN EL MISMO LABERINTO SIEMPRE, AUNQUE SEA EN UN LUGAR DIFERENTE CADA VEZ, DEBERA REVISAR LA INFORMACION CONTENIDA EN SU MEMORIA, DESCRIMINANDO LA QUE NO LE SIRVA, Y UNIENDO ESTA A LA NUEVA INFORMACION ADQUIRIDA PARA CREAR NUEVAS RUTAS.

CADA NUEVA RUTA SERA GUARDADA EN LA MEMORIA DE LA MAQUINA. DE ESTA MANERA CADA SALIDA DE UN MISMO LABERINTO SERA UN NUEVO APRENDIZAJE. SI EN LAS SUCESIVAS RESOLUCIONES EL APARATO SE ENCUENTRA CON QUE HAY DOS O MAS RUTAS QUE COMPARTEN CIERTO TRAMO DE LA SOLUCION, EL SIMULADOR DEBERA SER CAPAZ DE ENCADENAR ESTAS SOLUCIONES; O DICHO DE OTRA MANERA, DEBERA GUARDAR SOLO UNA SOLA VEZ LOS PEDAZOS COMPARTIDOS, OPTIMIZANDO ASI EL USO DE SU MEMORIA.

COMO SE VE, EL PROBLEMA RESULTA INTERESANTE, PUES CONFORME LO VAYAMOS ANALIZANDO NOS DAREMOS CUENTA DE QUE EXISTEN SITUACIONES DIVERSAS QUE DEBERAN SER TOMADAS EN CUENTA PARA TENER EXITO EN NUESTRA EMPRESA.

CON LO PLANTEADO HASTA AQUI, PARECE SER QUE EL PROBLEMA SE PUEDE RESOLVER MUY FACILMENTE. PERO HAY QUE TENER EN CUENTA QUE NO SE ESTA HABLANDO DE UN SER HUMANO, CON UNA CAPACIDAD DE PERCEPCION INFINITAMENTE GRANDE, Y UNA FACILIDAD INNATA PARA RELACIONAR LA INFORMACION ADQUIRIDA, SINO DE UNA MAQUINA CON LIMITACIONES MUY GRANDES EN AMBOS PROCEDIMIENTOS. SIN EMBARGO, SABEMOS QUE NINGUN PROCESO DE INTELIGENCIA ARTIFICIAL ES SENCILLO, POR TANTO SE ESCOGIO ESTE PROBLEMA POR PARECERNOS EL MAS SIMPLE, PUES EN EL ESTAN INVOLUCRADAS TAN SOLO UNAS CUANTOS PARAMETROS ESCENCIALES, QUE SON:

- STATUS
- DIRECCION
- POSICION RELATIVA
- DISTANCIA

ADICIONALMENTE, DURANTE LA RESOLUCION DEL PROBLEMA, IRAN SURGIENDO OTRAS VARIABLES AUXILIARES, QUE SERA NECESARIO USAR PARA PODER MANEJAR TODA LA INFORMACION.

TENIENDO UN PROBLEMA SENCILLO, SERA MAS FACIL NO PERDER EL CONTROL DEL MISMO. ADEMAS, COMO ILUSTRACION DEL CONCEPTO ES EXCELENTE, PUES NOS MUESTRA COMO DEBE REALIZARSE UN PROCESO DE APRENDIZAJE ORDENADO, LO CUAL ES BASICO EN CUALQUIER INVESTIGACION DE INTELIGENCIA ARTIFICIAL.

SABEMOS, POR OTRA PARTE, QUE EN CUALQUIER PROCESO DE APRENDIZAJE DE UN SER HUMANO, NO IMPORTA QUE TANTO ORDEN SE TRATE DE LLEVAR, SIEMPRE SURGIRAN VARIANTES O IDEAS EXPONTANEAS EN LA MENTE, QUE MODIFICARAN EL CURSO DE LOS SUCESOS. ESTO DEBERA ESTAR REPRESENTADO DE ALGUNA MANERA EN LA TOMA DE DECISIONES DEL SIMULADOR, POR MEDIO DE NUMEROS RANDOM.

RESTRICCIONES:

EL SER QUE SE ESTA SIMULANDO, PUEDE TENER CUALQUIER FORMA IMAGINABLE, PERO DESDE LUEGO TENDRA UNA PARTE FRONTAL, TRASERA, Y DOS LADOS. LO IMPORTANTE ES QUE DEBERA DE SER CAPAZ DE PERCIBIR LOS CAMBIOS QUE EL LABERINTO TENGA. EN OTRAS PALABRAS; LA COMPUTADORA DEBE SABER DE ALGUNA MANERA SI TIENE LA POSIBILIDAD DE CAMINAR PARA ENFRENTE, PARA ATRAS, O PARA CUALQUIERA DE LOS LADOS, Y A ESTO, ES A LO QUE LLAMAREMOS 'STATUS'. SIN EMBARGO CASI NINGUN SER ES CAPAZ DE DARSE CUENTA DEL STATUS EN QUE SE ENCUENTRAN SUS CUATRO LADOS. EL HOMBRE EN PARTICULAR, PUEDE VER SIN TENER QUE VOLTEAR, EL FRENTE Y SUS DOS LADOS. POR ESTA RAZON EL SER QUE SIMULAREMOS, TENDRA TAN SOLO TRES SENSORES, UNO AL FRENTE, Y UNO A CADA LADO; DE MANERA QUE SI DESEA VER EL STATUS EN QUE SE ENCUENTRA SU 'ESPALDA', DEBERA GIRAR SU CUERPO POR LO MENOS 90 GRADOS, DE MANERA QUE EL SENSOR DE UNO DE SUS LADOS PUEDA PERCIBIR ESTA SITUACION. ESTOS TRES SENSORES, SERAN LOS UNICOS MEDIOS CON LOS QUE CONTARA NUESTRO SER PARA OBTENER INFORMACION DEL EXTERIOR.

SABEMOS QUE NUESTRO SER CAMINARA SIEMPRE EN LINEA RECTA, POR TANTO DEBERA PODER MEDIR DE ALGUNA FORMA, LA DISTANCIA QUE HA RECORRIDO DE UN PUNTO A OTRO. ESTA SERA LA INFORMACION ADICIONAL QUE DEBERA OBTENER DEL MEDIO PARA PODER TRAZAR SUS RUTAS. PODEMOS IMAGINAR QUE ES UN CONTADOR QUE VA CONTANDO LAS VUELTAS QUE DA UNA RUEDA AL IR CAMINANDO.

POR TANTO, LA PRIMERA RESTRICCION QUE SE TENDRA, ES QUE LOS PASILLOS DEL LABERINTO QUE SE TRATE DE RESOLVERSE, DEBERAN TENER, FORSOZAMENTE ANGULOS DE 90 GRADOS ENTRE SI, PUES RESULTARIA DEMASIADA COMPLEJA LA RESOLUCION DEL MISMO, SI DAMOS LA POSIBILIDAD DE ANGULOS INTERMEDIOS.

LA SEGUNDA RESTRICCION, ES QUE LOS PASILLOS DEL LABERINTO DEBERAN TENER UNA ANCHURA CONSTANTE, DE MANERA QUE EL SER PUEDA TOMAR LAS PAREDES COMO UNA GUIA PARA PODER CAMINAR EN LINEA RECTA. PARA FACILITARNOS LA TAREA CONSIDERAREMOS QUE TIENEN UNA ANCHURA DE UN METRO.

TERCERO: LA DISTANCIA ENTRE UNA PUERTA, Y LA SIGUIENTE DEBERA ESTAR DADA EN UNIDADES CERRADAS. EN OTRAS PALABRAS; SI EL SER SE ENCUENTRA EN UNA PUERTA, Y EMPIEZA A CAMINAR, NO PUEDE ENCONTRARSE OTRA PUERTA A UNA DISTANCIA DE 10.5, U 25.34 METROS, SINO A 15 O 24 O 36 METROS. ESTO SE HACE CON EL PROPOSITO DE FACILITAR EL REGISTRO DE LAS POSICIONES POR LAS CUALES HAYA PASADO NUESTRO SER. POR TANTO LA DISTANCIA MINIMA QUE SE RECORRERA ANTES DE ENCONTRAR UN CAMBIO, DEBERA SER DE UN METRO.

DEBIDO A QUE LO QUE SE VA A REALIZAR NO ES SINO UNA SIMULACION, ENTONCES SE TENDRA QUE REEMPLAZAR LOS SENSORES. POR TANTO, LOS OPERADORES HARAN LAS VECES DE SENSORES Y DE CONTADORES DE LA DISTANCIA RECORRIDA, ES DECIR, SERAN LA PARTE MECANICA DEL SER. ADEMAS, COMO EL SIMULADOR NO SERA CAPAZ DE DARSE CUENTA CUANDO A LOGRADO SALIR DEL LABERINTO, SE LE TENDRA QUE DECIR DE ALGUNA FORMA.

POR OTRO LADO, EL SIMULADOR DEBERA PODER DAR VUELTAS A LA DERECHA Y A LA IZQUIERDA, Y CAMINAR LOS METROS QUE EL DECIDA CONVENIENTES. ESTO TAMBIEN ESTARA SUSTITUIDO POR LA PANTALLA DE LA COMPUTADORA, EN LA CUAL NOS DIRA HACIA DONDE DESEA DAR VUELTA, Y CUANTOS METROS DESEA CAMINAR.

EL LABERINTO QUE SE ESTE RESOLVIENDO PODRA TENER CUALQUIER NUMERO DE POSIBLES SALIDAS, O DE RUTAS A SEGUIR. SUS DIMENSIONES SERAN ILIMITADAS, QUEDANDO COMO UNICA RESTRICCION EN ESTE SENTIDO, LA MEMORIA CON LA CUAL CUENTA LA MAQUINA. ADEMAS POR LOGICA, PONDREMOS UN VALOR MAXIMO DE METROS QUE SE PUEDAN RECORRER EN LINEA RECTA, PUES SABEMOS QUE EL CONTADOR TENDRIA UN LIMITE DETERMINADO. PODRIA PENSARSE QUE ESTO FACILITA DE ALGUNA MENERA LA SOLUCION, PERO LA REALIDAD ES QUE NO TIENE NINGUNA IMPORTANCIA EL QUE LAS DISTANCIAS SEAN MUY GRANDES O NO, PUESTO QUE PARA UNA COMPUTADORA ES INDIFERENTE LA MAGNITUD DE LOS NUMEROS. AUN ASI, PARA QUE EL LABERINTO NO ESTE LIMITADO DE NINGUNA FORMA, EL NUMERO MAXIMO QUE PONDREMOS SERA DE 9,999 METROS .

RESUMEN DE CARACTERISTICAS DEL SIMULADOR

A) CARACTERISTICAS DE APRENDIZAJE:

- APRENDIZAJE DE RUTA DE SALIDA
- RELACION DE RUTAS ANTERIORES CON LAS NUEVAS
- MEMORIZACION COMPARTIDA DE LAS RUTAS
- TOMA DE DECISIONES ORDENADA Y/O EXPONTANEA

B) CANALES DE ENTRADA DE INFORMACION:

- TRES SENSORES DE STATUS.
- UN CONTADOR DE VUELTAS (METROS).

C) CANALES DE SALIDA DE INFORMACION:

- SEGUIR DE FRENTE O VUELTA A LA DERECHA O A LA IZQUIERDA
- METROS A RECORRER

D) RESTRICCIONES DEL LABERINTO:

- ANGULOS DE VUELTA DE 90 GRADOS
- ANCHO DE LOS PASILLOS IGUAL A LA UNIDAD (METRO)
- DIMENSIONES MULTIPLOS DE LA UNIDAD
- DISTANCIA MAXIMA EN LINEA RECTA DE 9,999 METROS
- NUMERO ILIMITADO DE RUTAS O SALIDAS

5.- ANALISIS Y SOLUCION

PARA EMPEZAR, COLOCAREMOS EL SIMULADOR EN UN PUNTO CUALQUIERA DE UN LABERINTO. EN ESE MOMENTO EL SIMULADOR DEBE SABER QUE ESTA EMPEZANDO. ENTONCES DESIGNARA A ESTA POSICION COMO EL ORIGEN, ES DECIR, SU POSICION EN EL EJE DE LAS "X" Y EN EL EJE DE LAS "Y" SERA CERO. EN LOS PUNTOS SIGUIENTES EL SIMULADOR SABRA SU POSICION CON RESPECTO AL ORIGEN, AYUDADO POR DOS VARIABLES, POS_X, Y POS_Y, LAS CUALES LE INDICARAN EN TODO MOMENTO EN QUE PUNTO DEL LOS EJES COORDENADOS SE ENCUENTRA.

ADEMAS SABRA QUE PARA LLEGAR A ESA POSICION, LA DISTACIA QUE TUVO QUE RECORRER ES DE CERO METROS (DESIGNAMOS METROS COMO LA UNIDAD DE LONGITUD, PARA FACILITAR LA EXPLICACION).

POR OTRA PARTE, DEBE DE ESCOGER UNA CIERTA ORIENTACION, ES DECIR, EL LADO HACIA EL CUAL ESTE APUNTANDO SU PARTE FRONTAL LA DEBE DE ASOCIAR A UN EJE DETERMINADO. EN LA PRACTICA ASIGNAREMOS ESTE EJE COMO EL "+X", LO CUAL SIGNIFICA QUE SI EL SIMULADOR EMPIEZA A CAMINAR HACIA EL FRENTE, LA DISTANCIA QUE RECORRA SE DEBERA IR AGREGANDO EN EL SENTIDO POSITIVO DE ESTE EJE. EN PUNTOS SIGUIENTES EL SIMULADOR PODRA ESTAR ORIENTADO HACIA CUALQUIERA DE LOS EJES CARTESIANOS, LOS CUALES DESIGNAREMOS DE LA SIGUIENTE MANERA.

ORIENTACION EN EL SENTIDO POSITIVO DE LAS X's = +X
ORIENTACION EN EL SENTIDO NEGATIVO DE LAS X's = -X
ORIENTACION EN EL SENTIDO POSITIVO DE LAS Y's = +Y
ORIENTACION EN EL SENTIDO NEGATIVO DE LAS Y's = -X

EN EL SIGUIENTE MOMENTO EL SIMULADOR LO QUE HARA, SERA REVISAR SU STATUS, ES DECIR, POR MEDIO DE SU INTERFASE CON EL MEDIO EXTERIOR, SE DARA CUENTA DE CUALES DE SUS TRES LADOS ESTAN ABIERTOS Y CUALES CERRADOS. DE ESTA MANERA CUANDO EL SIMULADOR REGRESE DE LA RUTINA DE INTERFASE, SE ENCONTRARA CON QUE SU STATUS ESTA DE LA SIGUIENTE MANERA:

POSIBILIDADES

LADO DERECHO	=	0	o 1
PARTE FRONTAL	=	0	o 1
LADO IZQUIERDO	=	0	o 1

DONDE LA FORMA EN QUE ESTARAN ORDENADOS SERA DERECHO,FRENTE,IZQUIERDO.

COMO NECESITAMOS MEMORIZAR LAS RUTAS PARA USARLAS EN FUTURAS BUSQUEDAS, LO MAS ADECUADO ES QUE DUPLIQUEMOS EL STATUS, DE MANERA QUE TENGAMOS UN STATUS_1 Y UN STATUS_2, DONDE EL PRIMER STATUS, PERMANECERA INALTERABLE DURANTE LA BUSQUEDA, Y EL SIGUIENTE, SERA DINAMICO DEBIDO A QUE COOPERARA DE MANERA PRIMORDIAL PARA LA RESOLUCION DEL LABERINTO, POR LO CUAL, AL FINAL DE LA RESOLUCION, LOS PUNTOS POR LOS CUALES PASE EL SIMULADOR, PODRAN TENER LOS STATUS DIFERENTES.

PARA EL STATUS_1 TENDREMOS LOS PARAMETROS D1,F1,I1
PARA EL STATUS_2 TENDREMOS LOS PARAMETROS D2,F2,I2

UNA VEZ QUE SE TIENEN LOS STATUS, PROCEDERA EL SIMULADOR A LA TOMA DE DECISION, DONDE COMO PRIMER PASO DEBERA REVISAR QUE NO SE ENCUENTRA EN UN CALLEJON SIN SALIDA (ESTE PROCESO SE DESCRIBIRA DESPUES).

EN CASO DE NO ENCONTRARSE EN UN CALLEJON SIN SALIDA, EL SIMULADOR DEBERA DECIDIR HACIA QUE LADO DESEA DAR LA VUELTA, O SI DESEA SEGUIR DE FRENTE. ESTO SE HACE BASANDOSE EN EL STATUS_2, EL CUAL LE INDICARA HACIA QUE LADOS PUEDE DIRIGIRSE. EL SIMULADOR TENDRA LAS SIGUIENTES PRIORIDADES:

A.- EN CASO DE SOLO ENCONTRAR UNO DE LOS TRES LADOS ABIERTO, EL SIMULADOR TENDRA QUE DIRIGIRSE FORZOSAMENTE EN ESA DIRECCION.

B.- SEGUIR DE FRENTE SIEMPRE QUE SE PUEDA.

C.- SI SE TIENE QUE DECIDIR ENTRE IR A LA IZQUIERDA O A LA DERECHA SE HARA GENERANDO UN NUMERO RANDOM QUE IRA DEL 1 AL 10, DONDE TODOS LOS NUMEROS MENORES O IGUALES A 5 HARAN QUE EL SIMULADOR TOME UNA DIRECCION Y VICEVERSA.

A LA DECISION QUE TOME EL SIMULADOR LE LLAMAREMOS SIMPLEMENTE, VUELTA, PUES AUNQUE SEGUIR DE FRENTE NO ES DAR LA VUELTA, PARA EL SIMULADOR ES INDISTINTO. ESTA INDICACION DE VUELTA SE MANDARA POR MEDIO DE LA RUTINA DE INTERFASE, PARA QUE EL SIMULADOR EJECUTE LA DECISION.

UNA VEZ QUE EL SIMULADOR A DECIDIDO HACIA QUE LADO DESEA IR, DEBE DE SABER CUAL SERA SU ORIENTACION DESPUES DE DAR LA VUELTA.
CON EL OBJETO DE NO CONFUNDIR TERMINOS, LLAMAREMOS DIRECCION A LA ORIENTACION CON QUE QUEDA EL SIMULADOR DESPUES DE DAR VUELTA. DESDE LUEGO ES EVIDENTE QUE LA DIRECCION SERA IGUAL A LA ORIENTACION, EN CASO DE QUE EL SIMULADOR DECIDA SEGUIR DE FRENTE.
PARA OBTENER LA DIRECCION, RECURRIMOS A LA SIGUIENTE TABLA:

TABLA 1

ORIENTACION	VUELTA	DIRECCION
CUALQUIERA	F	IGUAL
+X	D	-Y
+X	I	+Y
-X	D	+Y
-X	I	-Y
+Y	D	+X
+Y	I	-X
-Y	D	-X
-Y	I	+X

LA DIRECCION EN LA CUAL QUEDA EL SIMULADOR, ES MUY IMPORTANTE, PUES AL LLEGAR AL SIGUIENTE PUNTO, LA ORIENTACION DEL MISMO SERA IGUAL A LA DIRECCION QUE SE HAYA ENCONTRADO, Y DE ESTA MENERA SE ENCADENAN LOS PUNTOS QUE SE VAYAN ENCONTRANDO.

DESPUES DE DAR LA VUELTA, EL SIMULADOR EMPEZARA A CAMINAR, Y CADA

METRO QUE RECORRA REVISARA SU STATUS, Y VERIFICARA QUE ESTE SEA UN PUNTO SIGNIFICATIVO, ES DECIR, QUE TENGA UNA BIFURCACION, O BIEN QUE TENGA QUE DAR VUELTA. SI NO ES ASI, IRA INCREMENTANDO LA VARIABLE DE DISTANCIA, LA CUAL AL EMPEZAR A CAMINAR SE ENCONTRABA EN CERO.

AL MOMENTO DE ENCONTRARSE CON UN PUNTO SIGNIFICATIVO, LO PRIMERO QUE HARA EL SIMULADOR, SERA OBTENER LA ORIENTACION, LA CUAL COMO YA DIJIMOS ANTES ES IGUAL A LA DIRECCION DEL PUNTO DESDE EL CUAL PARTIO.

EN BASE A LA DISTANCIA RECORRIDA, LA ORIENTACION Y LA POSICION ANTERIOR, EL SIMULADOR DEBERA ENCONTRAR LA NUEVA POSICION EN QUE SE ENCUENTRA, LO CUAL SERA MUY SENCILLO, COMO EJEMPLIFICAREMOS:

```
POS_X(n-1)       = +10
POS_Y(n-1)       = -5

ORIENTACION (n) = -X
DISTANCIA (n)    =  8 METROS
```

COMO LA DISTANCIA SE RECORRIO HACIA EL EJE NEGATIVO DE LAS X'S, ENTONCES SE LE DEBE DE RESTAR LA DISTANCIA RECORRIDA A LA POS_X(n-1), PARA OBTENER LA POSICION POS_X(n). DONDE "n" ES UN CONTADOR QUE INDICA CUANTOS PUNTOS LLEVA REGISTRADOS EL SIMULADOR.

```
POS_X(n)         = +2
POS_Y(n)         = -5
```

AL LLEGAR A ESTE PUNTO, HEMOS COMPLETADO UN CIRCULO, PUES NOS ENCONTRAMOS EN LAS MISMAS CONDICIONES QUE CUANDO EMPEZAMOS, ES DECIR, QUE CONOCEMOS LA DISTANCIA, LA POSICION Y LA ORIENTACION. LO QUE RESTA ES EMPEZAR DE NUEVO, LLENDO A REGISTRAR EL STATUS EN EL CUAL SE ENCUENTRA EL SIMULADOR EN ESTA MOMENTO.

HASTA LO QUE HEMOS EXPLICADO AQUI, LO PODEMOS ENGLOBAR COMO UN PROCESO SIMPLE, PUES DE ESTA MANERA EL SIMULADOR ES CAPAZ DE RESOLVER LABERINTOS QUE NO TENGAN COMPLICACIONES.

RESUMIENDO EL PROCESO PARA TODOS LOS PUNTOS:

PROCESO SIMPLE:

1.- REGISTRAR LA DISTANCIA.
2.- CALCULAR LA POSICION EN BASE A LA DISTANCIA, POSICION ANTERIOR Y ORIENTACION.
3.- REGISTRAR LOS STATUS_1 Y STATUS_2 .
4.- ORIENTACION (n) = DIRECCION (n-1)
* 5.- REVISAR SI EL PUNTO ENCONTRADO NO ES UN CALLEJON SIN SALIDA.
6.- EN BASE AL STATUS_2 TOMAR LA DECISION (VUELTA)
7.- OBTENER LA DIRECCION
8.- REGRESAR AL PUNTO 1

* UN CALLEJON SIN SALIDA ES UN PUNTO EN EL LABERINTO, QUE COMO SU NOM

BRE LO INDICA, NO DA LA POSIBILIDAD DE SEGUIR DE FRENTE O DAR VUELTA A ALGUNO DE LOS LADOS. ENTONCES LA UNICA MANERA DE SALIR DE EL, ES REGRESAR POR EL CAMINO QUE SE TOMO PARA LLEGAR A EL.

EL SIMULADOR SE DARA CUANTA DE QUE ESTA EN UN CALLEJON SIN SALIDA CUANDO SE ENCUENTRE QUE SU STATUS_2 ES IGUAL A "000". EN ESE CASO LO QUE DEBE DE HACER ES REGRESARSE A LA POSICION ANTERIOR Y CERRAR LA ENTRADA POR LA CUAL LLEGO AL CALLEJON SIN SALIDA.

PROCEDIMIENTO:

1.- REVISAR QUE ES STATUS_2 SEA IGUAL A "000"

- SI NO LO ES, IR AL PUNTO SEIS DEL PROCESO SIMPLE.
- SI ES UN CALLEJON, PROCEDER AL PASO DOS.

2.- SE BORRARA LA INFORMACION QUE SE HAYA GENERADO AL LLEGAR AL CALLEJON. ES DECIR, LOS STATUS, LA ORIENTACION, Y LA POSICION. LO UNICO QUE SE SALVARA SERA LA DISTANCIA.

3.- RECORRERA EN REVERSA LA DISTANCIA QUE SE HAYA CAMINADO PARA LLEGAR AL CALLEJON, CON LO CUAL EL SIMULADOR ESTARA EN LA POSICION "n-1", LA CUAL A PARTIR DE ESTE MOMENTO SERA LA POSICION "n"

4.- UNA VEZ EN ESTA POSICION, EL SIMULADOR DEBERA REORIENTARSE, ES DECIR COLOCARSE EN LA MISMA POSICION QUE CUANDO LLEGO A ESTE PUNTO, QUE EN ESTE MOMENTO YA NO ES EL PUNTO "n-1", SINO EL PUNTO "n". LA REORIENTACION SE LLEVARA A CABO ASI:

- SI EL SIMULADOR HABIA DADO UNA VUELTA A LA IZQUIERDA CUANDO ENTRO EN EL CALLEJON, DEBERA DAR LA VUELTA A LA DERECHA PARA QUEDAR REORIENTADO, Y VICEVERSA.

- SI EL SIMULADOR SE HABIA SEGUIDO DE FRENTE, SE QUEDARA EN LA MISMA ORIENTACION, YA QUE ESTA NO VARIO AL MOMENTO DE ENTRAR AL CALLEJON.

5.- CERRAR LA ENTRADA POR LA CUAL SE ENTRO PARA LLEGAR AL CALLEJON. LO CUAL SE LOGRA PONIENDO UN CERO EN LA POSICION CORRESPONDIENTE A LA VUELTA. SI SE DIO LA VUELTA A LA DERECHA, SE DEBERA PONER UN CERO EN EL SENSOR QUE CORRESPONDE AL LADO DERECHO, EN EL STATUS_2.

SI VUELTA = F --) F2=0
SI VUELTA = D --) D2=0
SI VUELTA = I --) I2=0

ASI QUEDA ILUSTRADO EL HECHO DE QUE MIENTRAS EL STATUS_1 NO VARIE EL STATUS_2 LO HACE DE ACUERDO A LAS NECESIDADES QUE SE TENGAN.

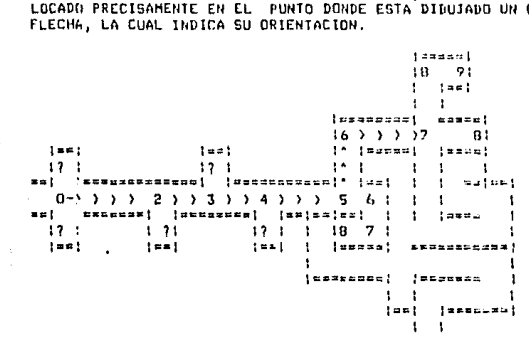
6.- POR ULTIMO BORRARA LA VUELTA, ASI COMO LA DIRECCION DE ESTE PUNTO, PUES AL LLEGAR A ESTE, TENDRA QUE TOMAR LA DECISION DE NUEVO.

COMO CABE LA POSIBILIDAD DE QUE PARA ENTRAR AL CALLEJON, EL SIMULADOR LO HAYA HECHO DESDE UN PUNTO DE OPCION UNICA, AL MOMENTO DE CERRAR LA ENTRADA DE ACCESO AL CALLEJON NOS ENCONTRAREMOS CON QUE ESTE PUNTO SE

HA CONVERTIDO EN UN CALLEJON SIN SALIDA, ENTONCES, ANTES DE SALIR DE ESTA RUTINA EL SIMULADOR DEBERA IR A REVISAR DE NUEVO SI EL PUNTO EN EL CUAL SE ENCUENTRA NO SE HA CONVERTIDO EN OTRO CALLEJON, ES DECIR, REGRESARA AL PASO UNO DE ESTE PROCEDIMIENTO.

EJEMPLO:

SE TIENE UN LABERINTO COMO EL ANTERIOR, DONDE EL SIMULADOR ESTA COLOCADO PRECISAMENTE EN EL PUNTO DONDE ESTA DIBUJADO UN CIRCULO CON UNA FLECHA, LA CUAL INDICA SU ORIENTACION.



EN EL PUNTO INICIAL EL LOS PARAMETROS DEL SIMULADOR SERAN LOS SIGUIENTES:

N	DISTAN	POS_X	POS_Y	STATUS_1 D1 F1 I1	STATUS_2 D2 F2 I2	ORIENT	VUELTA	DIRECC
1	0	0	0	1 1 1	1 1 1	+X	F	+X

EL CONTADOR ES IGUAL A UNO POR SER EL PUNTO DE ORIGEN, LA DISTANCIA Y LAS POSICIONES SON IGUAL A CERO. LA ORIENTACION CON QUE EMPIEZA EL SIMULADOR, LA CONSIDERA IGUAL A +X. EL PRIMER STATUS DE ESTE PUNTO NOS INDICA QUE SE PUEDE IR EN CUALQUIER DIRECCION QUE SE QUIERA. COMO EL SIMULADOR PUEDE IR AL FRENTE, DECIDE IR EN ESTA DIRECCION, POR LO CUAL LA DIRECCION QUE QUEDA ES IGUAL A LA ORIENTACION .

EL SIMULADOR RECORRE CUATRO METROS ANTES DE LLEGAR A UN PUNTO SIGNIFICATIVO, EL EL CUAL SE ENCUENTRA CON LAS SIGUIENTES CARACTERISTICAS:

2	4	4	0	1	1	0	1	1	0	+X	F	+X

EL CONTADOR SE INCREMENTA A DOS. LA DISTANCIA RECORRIDA SE ANOTA, Y SE SUMA EN EL SENTIDO POSITIVO DE LAS X'S, PUES LA ORIENTACION DEL SIMULADOR CORRESPONDE A LA DIRECCION DEL PUNTO ANTERIOR, LA CUAL ERA +X.

EL SIMULADOR TIENE LA POSIBILIDAD DE IR A LA IZQUIERDA O SEGUIR DE FRENTE, ENTONCES DECIDE IR DE FRENTE CON LO CUAL LLEGA AL PUNTO TRES:

3	2	6	0	0	1	1	0	1	1	+X	F	+X

LA POSICION EN "Y" SIGUE SIENDO CERO, PUES EL SIMULADOR NO HA CAMINADO MAS QUE EN EL EJE DE LAS X's.

4	2	8	0	1	1	0	1	1	0	+X	F	+X
5	2	10	0	0	1	1	0	1	1	+X	F	+X

EN EL PUNTO 6, EL SIMULADOR SE ENCUENTRA CON QUE SOLO PUEDE DAR VUELTA A LA DERECHA, CON LO CUAL LA DIRECCION DE ESTE PUNTO SEGUN LA TABLA 1 ES IGUAL A -Y.

6	1	11	0	1	0	0	1	0	0	+X	D	-Y

EN EL PUNTO 7 EL SIMULADOR SE ENCUENTRA EN SITUACION SIMILAR A LA ANTERIOR, Y GUIADO POR LA TABLA 1, SU NUEVA DIRECCION ES EN EL EJE NEGATIVO DE LAS X's.

7	1	11	-1	1	0	0	1	0	0	-Y	D	-X

HASTA ESTE MOMENTO TODOS LOS STATUS_1 Y STATUS_2 HAN PERMANECIDO IGUALES, Y ADEMAS NO NOS HEMOS ENCONTRADO CON UN STATUS_2 IGUAL A "000". PERO EN EL PUNTO 8. EL SIMULADOR SE ENCUENTRA EN LA SIGUIENTE SITUACION.

8	1	13	-1	0	0	0	0	0	0	-X

SU STATUS_2 ES IGUAL A "000" CON LO CUAL EL SIMULADOR DEBE HACER LO SIGUIENTE:

- REGRESARSE UN METRO.
- BORRAR TODOS LOS DATOS DEL PUNTO 8.
- DECREMENTAR EL CONTADOR CON LO CUAL SE ENCONTRARA EN EL PUNTO 7
- COMO LA VUELTA QUE SE DIO FUE A LA DERECHA, PARA REORIENTARSE NECESITA DAR VUELTA A LA IZQUIERDA.
- ADEMAS DEBERA CERRAR EL PARAMETRO DEL STATUS_2 CORREGPONDIENTE A EL LADO HACIA EL CUAL DIO LA VUELTA, ES DECIR QUE D2=0.

ENTOCES EL PUNTO 7 QUEDARA COMO SIGUE:

7	1	11	-1	1	0	0	0	0	0	-Y

LO CUAL NOS INDICA QUE EL STATUS_1 YA NO ES IGUAL AL STATUS_2,ADEMAS, DESPUES DE LA CORRECCION QUE SE EFECTUO, EL STATUS_2 DE ESTE PUNTO ES IGUAL A "000" CON LO CUAL EL SIMULADOR DEBERA PROCEDER DE LA MISMA MANERA QUE CON EL PUNTO 8. ASI NO ENCONTRAMOS QUE EL SIMULADOR ESTA DE NUEVO EN EL PUNTO 6.

6	1	11	0	1	0	0	0	0	0	+X

COMO TAMBIEN SE CONVIERTIO EN CALLEJON. ENTONCES REGRESAMOS AL

PUNTO 5, DONDE DESPUES DE LA CORRECCION DEL PUNTO 6, EL SIMULADOR SOLO PUEDE DAR VUELTA A LA IZQUIERDA.

5	2	10	0	0	1	1	0	0	1	+X	I	+Y
6	2	10	2	1	0	0	1	0	0	+Y	D	+X
7	3	13	2	1	1	1	1	1	1	+X	F	+X
8	2	15	2	0	0	0	0	0	0	+X		

CALLEJON ---> REGRESO AL PUNTO 7
DESICION AL AZAR HACIA EL LADO IZQUIERDO

9	3	13	2	1	1	1	1	0	1	+X	I	+Y
8	2	13	4	1	0	0	1	0	0	+Y	D	+X
9	1	14	4	0	0	0	0	0	0	+X		

CALLEJON ---> REGRESO AL PUNTO 8

8	2	13	4	1	0	0	0	0	0	+X		

DE ESTA MANERA, EL SIMULADOR, PUEDE ENCONTRAR EL CAMINO POR EL METODO DE LA PRUEBA Y EL ERROR. Y PODRIAMOS SEGUIR SIMULANDO SU COMPORTAMIENTO EN EL PAPEL, HASTA SALIR DEL LABERINTO MOSTRADO EN LA FIGURA.

HASTA AQUI HA QUEDADO UNA SOLUCION MUY SIMPLE. Y EN EL MOMENTO QUE EL SIMULADOR LOGRE SALIR DEL LABERINTO, LA RUTA POR LA CUAL SALIO QUEDARA GUARDADA EN SU MEMORIA COMO UNA SUCESION DE PUNTOS EN LOS CUALES HA QUEDADO REGISTRADO LA DISTANCIA RECORRIDA Y LA DIRECCION TOMADA, QUE ES A FINAL DE CUENTAS LO QUE REALMENTE IMPORTA, PARA QUE EL SIMULADOR PUEDA SALIR DE NUEVO. ASI EL SIMULADOR A REALIZADO UN PROCESO DE APRENDIZAJE MAQUINAL.

SI BIEN EL SIMULADOR YA SERIA CAPAZ DE SALIR DE UN LABERINTO SIMPLE, AUN NO LE ES POSIBLE SALIR DE UN LABERINTO COMPLICADO.

POR LABERINTO COMPLICADO ENTEDEMOS AQUEL LABERINTO EN EL CUAL EXISTEN CAMINOS QUE PUEDAN CONFUNDIR AL SIMULADOR. ESTOS CAMINOS NO SON OTRA COSA QUE LOS LOOPS.

ANALISIS DE LOOPS:

QUE PASARIA SI EL SIMULADOR EN ESTAS CONDICIONES SE ENCONTRARA CON UNA RUTA QUE LO HICIERA DAR UN RODEO Y REGRESAR A UN PUNTO POR EL CUAL YA HUBIERA PASADO ? PODRIA DARSE EL CASO QUE EL SIMULADOR SE QUEDARA DANDO VUELTAS UNA Y OTRA VEZ, SIGUIENDO LOS MANDATOS DE LA LOGICA SIMPLE QUE SE HA ENCONTRADO PARA EL.

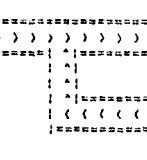
ENTONCES CONCLUIMOS QUE EL SIMULADOR NECESITA UN MEDIO PARA DARSE CUENTA [illegible] HA CAIDO EN UN LOOP, Y ESTO ES MUY SENCILLO, PUES EN CADA PUNTO CONTARA CON DOS REGISTRO EN LOS CUALES ESTA GUARDADA LA POSICION ASOCIADA A ESE PUNTO.

POR TANTO LO QUE SE TIENE QUE HACER PARA LOGRAR QUE EL SIMULADOR SE DE CUENTA QUE A REGRESADO A UN PUNTO POR EL CUAL YA PASO, ES COMPARAR LA POSICION DEL SIMULADOR EN EL ULTIMO PUNTO CON CADA UNA DE LAS POSICIONES DE LOS PUNTOS ANTERIORES, Y EN EL MOMENTO EN QUE LAS VARIBLES DE POSICION POS_X(n) Y POS_Y(n) COINCIDAN CON LA POSICION DE ALGUNO DEL LOS PUNTOS ANTERIORES, EL SIMULADOR ESTARA EN PRESENCIA DE UN LOOP.

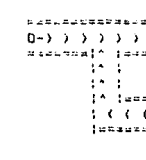
PARA PODER RESOLVER ESTA SITUACION, LO PRIMERO QUE TENEMOS QUE HACER ES UNA CLASIFICACION COMPLETA DE LOS LOOPS.

LA PRIMERA CLASIFICACION DE LOS LOOPS ES LA SIGUIENTE:

A.- LOOP CERRADO



B .- LOOP ABIERTO



DONDE LOOP CERRADO ES AQUEL CUYA CARACTERISTICA PRINCIPAL ES QUE LA RUTA QUE SE SIGUIO DESDE EL PUNTO INICIAL AL FINAL ES UNICA; ES DECIR QUE EL SIMULADOR NO TUVO OPORTUNIDAD DE ELEGIR EN EL MOMENTO EN QUE YA SE ENCONTRABA ADENTRO DEL LOOP. MIENTRAS QUE EL LOOP ABIERTO ES AQUEL QUE PRESENTA UNA RUTA NO UNICA; ES DECIR QUE EN LOS PUNTOS INTERMEDIOS EL SIMULADOR TUVO LA OPORTUNIDAD DE ELEGIR OTRA RUTA, ESTO ES, SE ENCONTRO MAS DE UNA ENTRADA, POR LO MENOS EN UNO DE ESTOS PUNTOS.

EN TERMINOS DE STATUS, PODEMOS DECIR QUE EN EL LOOP CERRADO LOS PUNTOS INTERMEDIOS SOLO PUEDEN PRESENTAR UNO DE ESTOS TRES STATUS: "001", "010" O BIEN "100"; QUE EN CUALQUIERA DE LOS TRES CASOS NOS REPRESENTAN PUNTOS EN LOS CUALES NO HAY MAS QUE UNA ELECCION POSIBLE PARA EL SIMULADOR.

ENTONCES LO QUE TIENE QUE HACER EL SIMULADOR PARA DISTINGUIR SI EL LOOP ES CERRADO O ABIERTO, ES REVISAR QUE TODOS LOS PUNTOS INTERMEDIOS CONTENGAN UN STATUS_2 IGUAL A LOS MOSTRADOS, Y SI ES ASI EN TODOS SUS PUNTOS ESTARA EN UN LOOP CERRADO, Y SI NO EN UN LOOP ABIERTO. ESTO ES IMPORTANTE PORQUE EL TRATAMIENTO EN LOS DOS CASOS ES DIFERENTE.

LO QUE SE HARA EN EL CASO DE UN LOOP CERRADO, SERA REORIENTAR EL SIMULADOR COMO ESTABA EN EL PUNTO INICIAL, Y CERRAR LAS ENTRADAS POR LAS CUALES ESTE PUNTO TIENE ACCESO AL LOOP.

EN CAMBIO EN UN LOOP ABIERTO, COMO NO SABEMOS SI LA(S) ENTRADA(S) QUE ENCONTRAMOS EN LOS PUNTOS INTERMEDIOS NOS PUEDAN LLEVAR A LA SOLU-

CION, LO QUE TENEMOS QUE HACER, ES CERRAR LAS ENTRADAS POR LAS CUALES EL PUNTO INICIAL TIENE ACCESO AL LOOP, Y SEGUNDO, CERRAR TODAS LAS ENTRADAS DEL PUNTO FINAL CON EL OBJETO DE CONVERTIRLO EN UN CALLEJON, Y DE ESTA MANERA,LOGRAR QUE SE REGRESE Y REVISE LA(S) ENTRADA(S) EN LOS PUNTOS INTERMEDIOS.

PERO COMO LOS LOOPS EN LOS LABERINTOS NO SON SIEMPRE TAN SENCILLOS COMO LOS MOSTRADOS EN LAS FIGURAS, SE TIENEN QUE HACER OTRAS PRUEBAS CON OBJETO DE DIFERENCIAR MEJOR A UN LOOP CERRADO DE UN ABIERTO. Y ESTO ES PORQUE MUCHOS LOOPS QUE EN APARIENCIA SON ABIERTOS, DESPUES DE CIERTAS TRANSFORMACIONES SE PUEDEN CONVERTIR EN LOOPS CERRADOS, CON LO CUAL SE EVITA QUE EL SIMULADOR TENGA QUE IR A REVISAR CADA UNA DE LAS ENTRADAS ENCONTRADAS EN LOS PUNTOS INTERMEDIOS.

LA RAZON ES LA SIGUIENTE. UN LOOP ES UN CIRCULO, Y NO IMPORTA CUANTOS PASOS DE EL SIMULADOR PARA COMPLETAR EL LOOP, A FINAL DE CUENTAS EL SIMULADOR TENDRA QUE HABER COMPLETADO UN GIRO A FAVOR O EN CONTRA DE LAS MANECILLAS DEL RELOJ. DESPUES DE HABER COMPLETADO EL GIRO, PUEDE QUE HAYA MUCHOS PUNTOS INTERMEDIOS CON ENTRADAS ABIERTAS A AMBOS LADOS, PERO SIN EMBARGO, DEPENDIENDO DEL GIRO QUE SE HAYA TOMADO, SIEMPRE HABRA UN LADO DE SIMULADOR QUE ESTARA APUNTANDO HACIA LA PARTE DE ADENTRO DEL LOOP, Y OTRA A LA PARTE DE AFUERA. SI EL GIRO ES A FAVOR DE LAS MANECILLAS DEL RELOJ, EL LADO DERECHO DEL SIMULADOR SIEMPRE APUNTARA HACIA LA PARTE DE ADENTRO DEL LOOP, MIENTRAS QUE SI EL GIRO ES EN CONTRA DE LAS MANECILLAS DEL RELOJ, EL LADO IZQUIERDO SERA EL QUE APUNTE HACIA LA PARTE DE ADENTRO. SI EN EL CAMINO DE UN LOOP, EL SIMULADOR SE ENCUENTRA CON UNA ENTRADA QUE APUNTA HACIA LA PARTE DE ADENTRO DEL LOOP, EL NO LO PUEDE SABER, MAS EN EL MOMENTO QUE EL LOOP SE CIERRA, EL SIMULADOR CONOCE PERFECTAMENTE HACIA QUE LADO A COMPLETADO EL GIRO, ENTONCES TODAS LAS ENTRADAS QUE APUNTEN HACIA LA PARTE CENTRAL DEL LOOP, NO TIENEN PORQUE SER REVISADAS, PUESTO QUE SABEMOS QUE NO NOS PUEDEN CONDUCIR A NINGUN LUGAR FUERA DEL LOOP. SIN EMBARGO, LAS ENTRADAS QUE SE ENCUENTREN Y QUE APUNTAN HACIA LA PARTE EXTERNA DEL LOOP SI PUEDEN CONDUCIRNOS A LA SOLUCION, POR LO TANTO SON SUMAMENTE IMPORTANTES.

PARA SABER EN QUE SENTIDO SE HA COMPLETADO EL LOOP ES MUY SENCILLO. SE REVISAN TODOS LOS PUNTOS INTERMEDIOS, Y SE VAN SUMANDO LAS VUETAS A LA DERECHA Y A LA IZQUIERDA. AL TERMINAR, EL LADO HACIA EL CUAL SE HAYA COMPLETADO EL CIRCULO, SERA EL QUE CONTENGA LA CANTIDAD MAYOR. ES DECIR, SI LAS VUELTAS A LA DERECHA SON MAYORES QUE LAS VUELTAS A LA IZQUIERDA, EL GIRO SE COMPLETO A FAVOR DE LAS MANECILLAS DEL RELOJ Y VICEVERSA.

ENTONCES CUANDO SE ESTA EN PRESENCIA DE UN LOOP, SIN IMPORTAR SI ESTE ES CERRADO O ABIERTO, LAS ENTRADAS QUE APUNTEN HACIA LA PARTE CENTRAL DEL LOOP, NO TIENEN MAYOR TRASCENDENCIA. POR LO TANTO SE DEBE EVITAR QUE EL SIMULADOR LAS REVISE EN CASO DE QUE RESULTE SER ABIERTO. EL PROCEDIMIENTO SERA EL SIGUIENTE:

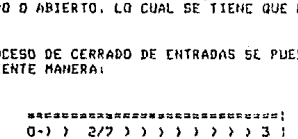
UNA VEZ QUE SE SABE HACIA QUE LADO SE HA COMPLETADO EL GIRO (DERECHO EN CASO DE MANECILLAS DEL RELOJ Y VICEVERSA), SI ESTE ES DERECHO, SE REVISARAN TODOS LOS PUNTOS INTERMEDIOS, Y LAS ENTRADAS DERECHAS QUE ESTEN ABIERTAS, SE CERRARAN, Y SI EL GIRO ES IZQUIERDO, TODAS LAS ENTRADAS IZQUIERDAS SE CERRARAN.

SI GIRO=DERECHO ==) CERRAR TODAS LAS F2 INTERMEDIAS
SI GIRO=IZQUIERDO ==) CERRAR TODAS LAS I2 INTERMEDIAS

SIN EMBARGO AL IR REALIZANDO ESTA OPERACION EN CADA PUNTO INTERMEDIO, PRIMERO SE DEBE REVISAR SI LA ENTRADA QUE SE VA A CERRAR NO ES UNICA, PUES EN CASO DE SERLO SE DEBE DE DEJAR ABIERTA, YA QUE EL STATUS DE ESTE PUNTO CORRESPONDERA AL DE UN LOOP CERRADO.

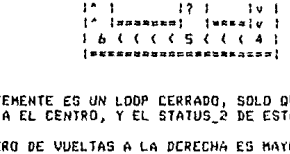
UNA VEZ REALIZADO LO ANTERIOR, PODREMOS ESTAR EN PRESENCIA DE UN LOOP CERRADO O ABIERTO, LO CUAL SE TIENE QUE REVISAR DE LA FORMA ANTES DESCRITA.

EL PROCESO DE CERRADO DE ENTRADAS SE PUEDE MOSTRAR GRAFICAMENTE DE DE LA SIGUIENTE MANERA:



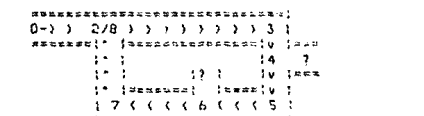
EVIDENTEMENTE ES UN LOOP CERRADO, SOLO QUE EN EL PUNTO 5 TIENE UNA ENTRADA HACIA EL CENTRO, Y EL STATUS_2 DE ESTE PUNTO ES "110".

EL NUMERO DE VUELTAS A LA DERECHA ES MAYOR QUE EL NUMERO DE VUELTAS A LA IZQUIERDA, EN LOS PASOS INTERMEDIOS, QUE EN ESTA CASO VAN DEL 3 AL 6, POR LO CUAL EL GIRO ES DERECHO. ENTONCES SE DEBEN DE CERRAR TODAS LAS ENTRADAS DERECHAS (D2) INTERMEDIAS, CON LO CUAL EN LA MEMORIA DEL SIMULADOR EL LOOP QUEDARA DE LA SIGUIENTE MANERA.

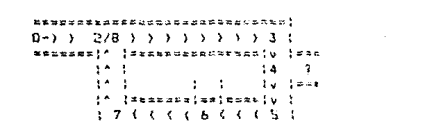


ES DECIR EL STATUS_2 DEL SIMULADOR QUEDO ASI "010", Y AL REVISAR TODOS LOS PUNTOS INTERMEDIOS, SE CONSIDERARA COMO UN LOOP CERRADO.

OTRO EJEMPLO:



AL MOMENTO DE APLICAR LA REGLA ANTERIOR, EL LOOP QUEDARA DE LA SIGUIENTE MANERA EN LA MEMORIA.

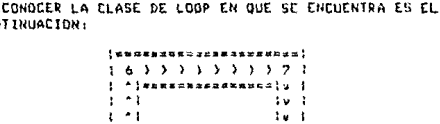


EL LOOP SIGUE SIENDO ABIERTO, PERO SIN EMBARGO, EL SIMULADOR AL REGRESARSE EVITARA REVISAR LA ENTRADA QUE ESTABA ABIERTA EN EL PUNTO 6, LO CUAL YA ES UN AHORRO EN TERMINOS DE ENTRADAS A REVISAR.

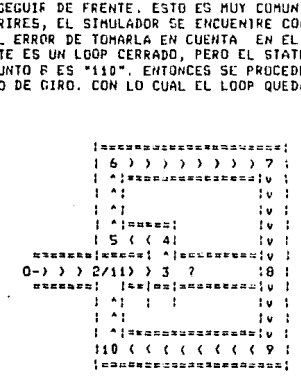
COMO SE VE SE TRATA SIEMPRE DE LOGRAR QUE LOS LOOPS SEAN CERRADOS, PUES ESTO NOS EVITA MUCHISIMAS REVISIONES.

UNA DE LAS VENTAJAS QUE SE PUEDEN PERDER AL CERRAR LAS ENTRADAS, ES QUE SI EL LOOP FUE ABIERTO, Y LA SOLUCION AL LABERINTO RODEA AL LOOP, PODRIA DARSE EL CASO QUE DOS DE LAS ENTRADAS INTERNAS DEL LOOP ESTUVIERAN INTERCOMUNICADAS Y CORTARAN EL LOOP EN DOS, HACIENDO LA RUTA DE SALIDA UN POCO MAS RAPIDA.

OTRA DE LOS PROBLEMAS A QUE SE PUEDE ENFRENTAR EL SIMULADOR AL TRATAR DE RECONOCER LA CLASE DE LOOP EN QUE SE ENCUENTRA ES EL QUE SE ILUSTRA A CONTINUACION:



EL SIMULADOR AL LLEGAR AL PUNTO 3 SE DIO LA VUELTA A LA IZQUIERDA EN VEZ DE SEGUIR DE FRENTE. ESTO ES MUY COMUN CUANDO AL ESTAR REVISANDO RUTAS ANTERIRES, EL SIMULADOR SE ENCUENTRE CON ALGUNA RUTA PARECIDA, Y CAIGA EN EL ERROR DE TOMARLA EN CUENTA EN EL MOMENTO DE DECIDIR LA VUELTA. ESTE ES UN LOOP CERRADO, PERO EL STATUS_2 DEL PUNTO 3 ES "111", Y EL DEL PUNTO 8 ES "110". ENTONCES SE PROCEDE A AJUSTAR LOS STATUS, SEGUN EL TIPO DE GIRO. CON LO CUAL EL LOOP QUEDARA DE LA SIGUIENTE MANERA.



EL PUNTO 8 SE CONVERTIRA EN UN PASILLO, PERO EL STATUS_2 DEL PUNTO 3 QUEDARA ASI: "011", LO CUAL INDICA AUN QUE EL LOOP NO ES CERRADO. LO CUAL ES ERRONEO.

EL PROBLEMA SE RESOLVERA DETERMINANDO LAS CARACTERISTICAS DEL PUNTO 3.

GIRO = DERECHO
VUELTA = IZQUIERDA

COMO VEMOS, EL LOOP ES DE GIRO DERECHO, Y SI LA VUELTA HUBIERA SIDO A LA DERECHA, LA PARTE FRONTAL NUNCA HUBIERA TENIDO LA POSIBILIDAD DE APUNTAR HACIA EL CENTRO DEL LOOP.

ENTONCES EL PROBLEMA SE RESUELVE MUY FACILMENTE, PUES CUANDO EL GIRO ES HACIA UN LADO DETERMINADO, Y LA VUELTA DADA EN UN PUNTO INTERMEDIO ES EN SENTIDO OPUESTO, LA PARTE FRONTAL SIEMPRE ESTA APUNTANDO HACIA EL CENTRO; LO UNICO QUE TENEMOS QUE REVISAR ES QUE SE CUMPLA ESTA CONDICION, Y DE SER ASI, CERRAR LA PUERTA FRONTAL, LA CUAL DE ANTEMANO PODRIA YA ESTAR CERRADA.

F2 = 0

DESPUES DE REALIZAR ESTAS PRUEBAS SE PUEDE SABER CON SEGURIDAD SI UN LOOP ES CERRADO O ABIERTO, Y ENTONCES SEGUN LA CLASE DE LOOP DE QUE SE TRATE SE PROCEDERA COMO SE INDICA A CONTINUACION.

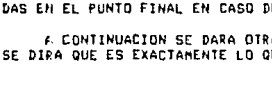
SE DIJO QUE CUANDO EL LOOP FUERA ABIERTO, EL SIMULADOR DEBERIA REGRESAR A REVISAR LAS ENTRADAS QUE ESTUVIERAN ABIERTAS. Y EN CASO DE SER CERRADO, SIMPLEMENTE SE TENDRIA QUE REORIENTAR EL SIMULADOR.

SE DIJO TAMBIEN QUE EN CUALQUIERA DE LOS DOS CASOS SE TENDRIAN QUE CERRAR CIERTAS ENTRADAS EN EL PUNTO DE INICIO DE LOOP, Y TODAS LAS ENTRADAS EN EL PUNTO FINAL EN CASO DE SER LOOP ABIERTO.

A CONTINUACION SE DARA OTRA CLASIFICACION DE LOS LOOPS POSIBLES, Y SE DIRA QUE ES EXACTAMENTE LO QUE SE DEBE REALIZAR EN CADA CASO.

ANALISIS DE LOOPS ABIERTOS:

CASO 1



CONDICIONES

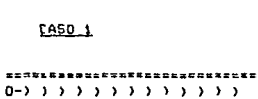
ORI (i) <> ORI (f)

DIR (i) <> ORI (f)

GIRO = DERECHO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
GIRO = DERECHO ==> D2(i) = 0
RUTINA DE CALLEJON

CASO 2



CONDICIONES

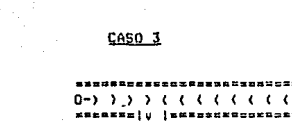
ORI (i) <> ORI (f)

DIR (i) <> ORI (f)

GIRO = IZQUIERDO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
GIRO = IZQUIERDO ==> I2(i) = 0
RUTINA DE CALLEJON

CASO 3



CONDICIONES

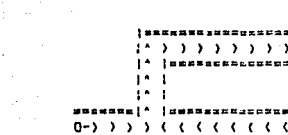
ORI (i) = -ORI (f)

DIR (i) () ORI (f)

GIRO = IZQUIERDO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = F1(i) = 0
GIRO = IZQUIERDO =⇒ D2(i) = 0
RUTINA DE CALLEJON

CASO 4



CONDICIONES

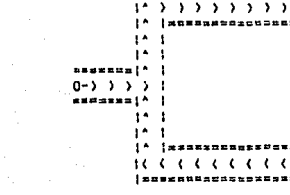
ORI (i) () -ORI (f)

DIR (i) () ORI (f)

GIRO = DERECHO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = F1(i) = 0
GIRO = DERECHO =⇒ I2(i) = 0
RUTINA DE CALLEJON

CASO 5



CONDICIONES

ORI (i) () ORI (f)

DIR (i) = ORI (f)

GIRO = DERECHO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
GIRO = DERECHO ==) D2(i) = 0
RUTINA DE CALLEJON

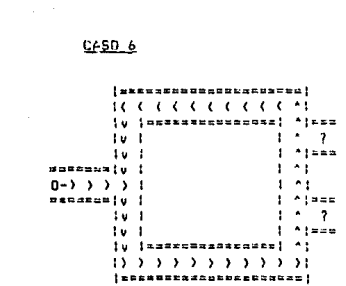
CASO 6

CONDICIONES

ORI (i) <> ORI (f)

DIR (i) = ORI (f)

GIRO = IZQUIERDO



PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
GIRO = IZQUIERDO ==) I2(i) = 0
RUTINA DE CALLEJON

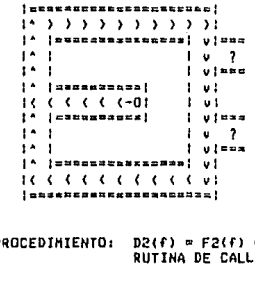
CASO 7

CONDICIONES

ORI (i) <> ORI (f)

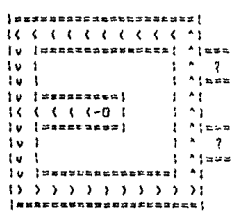
DIR (i) = ORI (f)

GIRO = DERECHO



PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
RUTINA DE CALLEJON

CASO 8



CONDICIONES

ORI (i) <> ORI (f)

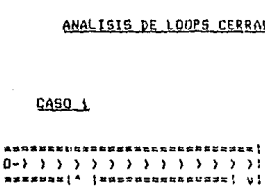
DIR (i) = ORI (f)

GIRO = IZQUIERDO

PROCEDIMIENTO: D2(f) = F2(f) = I2(f) = 0
RUTINA DE CALLEJON

ANALISIS DE LOOPS CERRADOS:

CASO 1



CONDICIONES

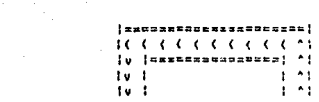
ORI (i) <> ORI (f)

DIR (i) <> ORI (f)

GIRO = DERECHO

PROCEDIMIENTO: GIRO = DERECHO =>) VUELTA A LA DERECHA
GIRO = DERECHO =>) D2(i) = F2(i) = 0
CONTADOR = 1
RUTINA DE CALLEJON

CASO 2



CONDICIONES

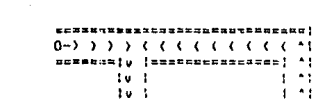
ORI (i) () ORI (f)

DIR (i) () ORI (f)

GIRO = IZQUIERDO

PROCEDIMIENTO: GIRO = IZQUIERDO = VUELTA A LA IZQUIERDA
GIRO = IZQUIERDO ==) I2(i) = F2(i) = 0
CONTADOR = i
RUTINA DE CALLEJON

CASO 3



CONDICIONES

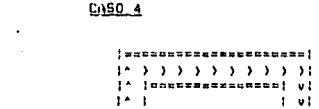
ORI (i) = -ORI (f)

DIR (i) () ORI (f)

GIRO = IZQUIERDO

PROCEDIMIENTO: GIRAR 180 GRADOS
F2(i) = 0
GIRO = IZQUIERDO ==) D2(i) = 0
CONTADOR = i
RUTINA DE CALLEJON

CASO 4



CONDICIONES

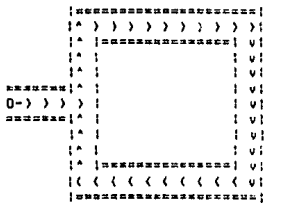
ORI (i) () -ORI (f)

DIR (i) () ORI (f)

GIRO = DERECHO

PROCEDIMIENTO: GIRAR 180 GRADOS
F2(i) = 0
GIRO = DERECHO ==) I2(i) = 0
CONTADOR = i
RUTINA DE CALLEJON

CASO 5



CONDICIONES

ORI (i) <> ORI (f)

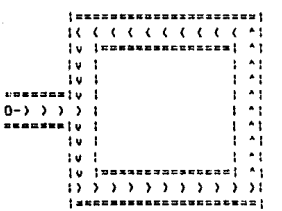
DIR (i) = ORI (f)

GIRO = DERECHO

VUELTA(i) = IZQUIERDA

PROCEDIMIENTO: D2(i) = F2(i) = I2(i) = 0
GIRO = DERECHO ========>!
VUELTA(i) = IZQUIERDA ==>! VUELTA A LA DERECHA
CONTADOR = i
RUTINA DE CALLEJON

CASO 6



CONDICIONES

ORI (i) <> ORI (f)

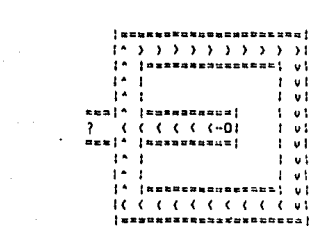
DIR (i) = ORI (f)

GIRO = IZQUIERDO

VUELTA(i) = DERECHA

PROCEDIMIENTO: D2(i) = F2(i) = I2(i) = 0
GIRO = IZQUIERDO <======!
VUELTA(i) = DERECHA ==>! VUELTA A LA IZQUIERDA
CONTADOR = i
RUTINA DE CALLEJON

CASO 7



CONDICIONES

ORI (i) <> ORI (f)

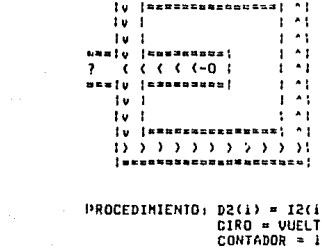
DIR (i) = ORI (f)

GIRO = DERECHO

VUELTA(i) = DERECHA

PROCEDIMIENTO: D2(i) = I2(i) = 0
GIRO = VUELTA(i) = DERECHA==> VUELTA A LA IZQUIERDA
CONTADOR = 1
RUTINA DE CALLEJON

CASO 8



CONDICIONES

ORI (i) <> ORI (f)

DIR (i) = ORI (f)

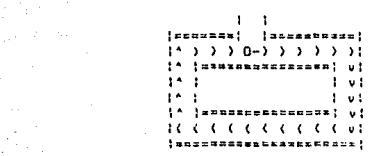
GIRO = IZQUIERDO

VUELTA(i) = IZQUIERDA

PROCEDIMIENTO: D2(i) = I2(i) = 0
GIRO = VUELTA(i) = IZQUIERDA==> VUELTA A LA DERECHA
CONTADOR = 1
RUTINA DE CALLEJON

ANALISIS DE LOOP EN POSICION INICIAL

CASO 1



CONDICIONES

ORI (i) = ORI (f)

i = 1

PROCEDIMIENTO: F2(i) = 0
CONTADOR = i
RUTINA DE CALLEJON

----------0----------

UNA VEZ CONCLUIDO EL ANALISIS DE LOS LOOPS, EL PROCESO SIMPLE QUE HABIAMOS DESCRITO, SE MODIFICARA PARA INCLUIR LA SOLUCION DE LOOPS.

PROCESO MODIFICADO:

1.- REGISTRAR LA DISTANCIA.
2.- CALCULAR LA POSICION EN BASE A LA DISTANCIA, POSICION ANTERIOR Y ORIENTACION.
3.- REGISTRAR LOS STATUS_1 Y STATUS_2 .
4.- ORIENTACION (n) = DIRECCION (n-1)
5.- REVISAR SI EL PUNTO ENCONTRADO NO ES UN CALLEJON SIN SALIDA.
6.- REVISAR SI EL SIMULADOR NO ESTA EN PRESENCIA DE UN LOOP. EN CASO DE SER ASI, RESOLVER EL LOOP Y REGRESAR AL PASO 5. SI NO, SEGUIR CON EL PASO 7
7.- EN BASE AL STATUS_2 TOMAR LA DICISION (VUELTA)
8.- OBTENER LA DIRECCION
9.- REGRESAR AL PUNTO 1

ESTE PROCESO YA NOS DA LA POSIBILIDAD DE RESOLVER CUALQUIER LABERINTO POR MUY COMPLICADO QUE ESTE SEA, SIN EMBARGO FALTA LA PARTE MAS IMPORTANTE DEL SISTEMA, QUE ES LA UTILIZACION DE LAS RUTAS APRENDIDAS

PARA FACILITARSE LA TAREA DE ENCONTRAR LA SALIDA DE UN LABERINTO DETERMINADO.

PARA PODER REALIZAR ESTE PROCESO, EL SIMULADOR GUARDARA TODOS LOS PUNTOS QUE LO HAYAN LLEVADO A RESOLVER EL LABERINTO, LE ASIGNARA UN NUMERO DE RUTA, Y EN LOS PROCESOS SIGUIENTES, SE REFERIRA A ESTA SOLUCION CON ESE NUMERO.

CADA VEZ QUE EL SIMULADOR ENCUENTRE UNA NUEVA RUTA PARA RESOLVER UN LABERINTO, LA GUARDARA EN LA MEMORIA, PARA PODER UTILIZARLA DE NUEVO. DE ESTA MANERA PODRA LLEGAR A TENER UN CONOCIMIENTO CASI TOTAL DE UN LABERINTO, DE MANERA QUE CUANTAS MAS VECES LO RESUELVA DESDE PUNTOS DE ORIGEN DISTINTOS CADA VEZ, SE FACILITARA LA SOLUCION MAS Y MAS.

PARA HACER USO DE LA INFORMACION GUARDADA EN LA MEMORIA, EL SIMULADOR RECORDARA LAS RUTAS RECORRIDAS CADA VEZ QUE SE QUIERA RESOLVER UNA NUEVA RUTA. DOS O MAS RUTAS PUEDEN COMPARTIR PEDAZOS DE LA SOLUCION; ES DECIR, QUE SI HABLAMOS DE LA SEGUNDA RUTA, PUEDE QUE ESTA TENGA UNA PARTE TOTALMENTE ORIGINAL, PERO QUE A LA MITAD DEL CAMINO SE ENCUENTRE CON LA PRIMERA RUTA, POR LO CUAL DESDE ESE MOMENTO HASTA EL FINAL, LAS DOS RUTAS COMPARTIRIAN UN NUMERO "N" DE PUNTOS EN COMUN. ENTONCES UNA DE LAS CARACTERISTICAS CON LAS CUALES DEBE CUMPLIR EL SIMULADOR ES LA DE NO GUARDAR INFORMACION DUPLICADA, O TRATAR DE QUE SEA LA MENOS POSIBLE.

PARA PODER RELACIONAR UNA NUEVA RUTA CON LAS ANTERIORES, EL SIMULADOR PROCEDERA DE LA SIGUIENTE MANERA.

DESDE EL PUNTO DE ORIGEN, EL SIMULADOR PRETENDERA ENCONTRAR ALGUN PUNTO QUE TENGA LAS MISMAS CARACTERISTICAS QUE ESTE. COMO EL SIMULADOR PUEDE QUEDAR ORIENTADO DE MANERA DIFERENTE CADA VEZ EN EL LABERINTO, LA ORIENTACION DE LAS RUTAS ANTERIORES NO TENDRA NINGUNA IMPORTANCIA.

EL PROCESO CONSTARA DE DOS FASES MUY IMPORTANTES:

- TOMA DE DECISION
- PREDICCION

DONDE DESDE LUEGO LA PREDICCION IMPLICA UNA TOMA DE DECISION.

EL SIMULADOR DEBERA REVISAR SI SE ENCUENTRA EN ALGUN PUNTO CONOCIDO CON ANTERIORIDAD, Y LO HARA RUTA A RUTA, SIGUIENDO EL ORDEN DE SU NUMERACION, EMPEZANDO POR LA RUTA 1 , RUTA 2, HASTA REVISAR CUANTAS RUTAS EXISTAN EN CASO DE SER NECESARIO. MAS SI EN LA PRIMERA RUTA SE ENCUENTRA UN PUNTO QUE LE PARESCA IGUAL, ENTONCES YA NO SEGUIRA REVISANDO Y TOMARA POR BUENO LO ENCONTRADO EN ESA RUTA.

LO QUE EL SIMULADOR TRATARA DE ENCONTRAR ES SI EXISTE UN PUNTO EN ALGUNA DE LAS RUTAS EXISTENTES CUYA DISTANCIA RECORRIDA PARA LLEGAR A EL, ASI COMO SU STATUS_1 ES EL MISMO QUE EN EL PUNTO EN EL CUAL SE ENCUENTRA. DE SER ASI, TOMARA COMO DECISION (VUELTA), AQUELLA QUE ESTE REGISTRADA EN EL PUNTO ENCONTRADO EN LAS RUTAS, PUESTO QUE SUPONDRA QUE YA

SE ENCUENTRA AHI. ESTO DESDE LUEGO PUEDE SER CIERTO O NO CUANDO HABLAMOS DE UN SOLO PUNTO, PERO EL CASO ES QUE SE TOMARA COMO SI LOS PUNTOS FUERAN EL MISMO.

DE ESTA MANERA, CUANDO EL SIMULADOR TERMINE LA RUTINA DE BUSQUEDA EN MEMORIA, YA TRAERA UNA DECISION, POR LO CUAL YA NO TENDRA QUE TOMARLA AL AZAR, COMO ANTERIORMENTE.

UNA VEZ QUE RECIBA LA INFORMACION DEL SIGUIENTE PUNTO, ENTONCES TRATARA DE ENCONTRAR DOS PUNTOS IDENTICOS EN LAS RUTAS, Y QUE ESTEN EN FORMA CONSECUTIVA, ES DECIR QUE BUSCARA EL PUNTO NUEVO, MAS EL PUNTO QUE YA HABIA ENCONTRADO ANTES. SI LOGRARA LOCALIZARLOS EN LAS RUTAS, TOMARA LA DECISION DEL ULTIMO PUNTO ENCONTRADO, Y REGRESARA.

POSTERIORMENTE EL SIMULADOR TRATARA DE ENCONTRAR 3 PUNTOS CONSECUTIVOS. Y LUEGO 4 ETC.

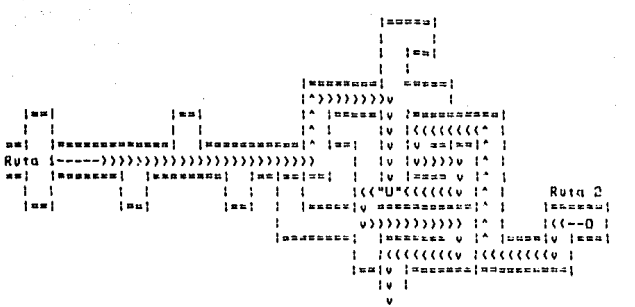
AL LLEGAR EL MOMENTO EN QUE EL SIMULADOR SE ENCUENTRE QUE YA TIENE 10 PUNTOS IDENTICOS EN LA MEMORIA Y EN LA RUTA NUEVA QUE ESTA ENCONTRANDO, CONSIDERARA QUE LA RUTA ENCONTRADA EN LA MEMORIA ESTA PLENEMENTE IDENTIFICADA, POR LO CUAL PROCEDERA NO SOLAMENTE A TOMAR LA DECISION DE LA MEMORIA, SINO QUE INTENTARA PREDECIR CUAL ES LA DISTANCIA QUE SE RECORRERA PARA LLEGAR AL SIGUIENTE PUNTO, ASI COMO EL STATUS QUE SE ENCONTRARA.

ES EN ESTE MOMENTO CUANDO SE COMPRUEBA QUE LA MAQUINA REALMENTE APRENDIO LAS RUTAS QUE HA RECORRIDO, PUES ES CAPAZ DE ASOCIAR DOS RUTAS CUYOS EJES COORDENADOS PUEDEN O NO, TENER LA MISMA ORIENTACION.

SI DESPUES DE LLEVAR UN NUMERO DETERMINADO DE PUNTOS IDENTICOS, EL SIMULADOR SE ENCUENTRA CON UN PUNTO QUE NO CONCUERDA CON LO PREVISTO, QUIERE DECIR QUE EL PUNTO DE LA RUTA EN EL QUE SUPONIA QUE ESTABA, REALMENTE NO ES ASI, POR LO QUE SE TENDRA QUE EMPEZAR LA BUSQUEDA DE NUEVO, Y SE HARA DE LA SIGUIENTE MANERA.

SI POR EJEMPLO EL SIMULADOR SE ENCONTRABA EN LA BUSQUEDA DE 15 PUNTOS, ENTONCES DECREMENTARA ESTA CANTIDAD Y TRATARA DE BUSCAR LOS 14 ULTIMOS PUNTOS. DE NO ENCONTRARLOS VOLVERA A DECREMENTAR Y BUSCARA LOS 13 ULTIMOS, HASTA QUE ENCUENTRE UN PUNTO O UNA SERIE DE PUNTOS EN LAS RUTAS QUE COINCIDAN CON LOS PUNTOS DE LA NUEVA RUTA. EN CASO DE QUE EL SIMULADOR LLEGE A DECREMENTAR A CERO, QUERRA DECIR QUE NI SIQUIERA EL ULTIMO PUNTO SE PUDO ENCONTRAR. POR LO CUAL SE DARA POR CONCLUIDA LA BUSQUEDA, Y SE DARA FIN A ESTA RUTINA, DEJANDOLA LISTA PARA LA REVISION DEL SIGUIENTE PUNTO.

EL DIAGRAMA DE DOS RUTAS QUE SE ENCONTRARAN PODRIA SER EL SIGUIENTE.



POR ULTIMO EL PROCESO COMPLETO TENIENDO EN CUENTA LA BUSQUEA EN MEMORIA SERIA EL SIGUIENTE:

PROCESO COMPLETO:

1.- REGISTRAR LA DISTANCIA.
2.- CALCULAR LA POSICION EN BASE A LA DISTANCIA, POSICION ANTERIOR Y ORIENTACION.
3.- REGISTRAR LOS STATUS_1 Y STATUS_2 .
4.- ORIENTACION (n) = DIRECCION (n-1)
5.- REVISAR SI EL PUNTO ENCONTRADO NO ES UN CALLEJON SIN SALIDA.
6.- REVISAR SI EL SIMULADOR NO ESTA EN PRESENCIA DE UN LOOP.
EN CASO DE SER ASI, RESOLVER EL LOOP Y REGRESAR AL PASO 5. SI NO, SEGUIR CON EL PASO 7
7.- BUSQUEDA EN MEMORIA. EN CASO DE ENCONTRAR ALGO TOMAR DECISION, O SI ES NECESARIO PREDECIR, Y SALTAR AL PASO 9. SINO SEGUIR CON EL PASO 8.
8.- EN BASE AL STATUS_2 TOMAR LA DECISION (VUELTA)
9.- OBTENER LA DIRECCION
10.- REGRESAR AL PUNTO 1

6.- DIAGRAMA DE FLUJO

INICIO

OLVIDO

INTERFASE

FIN

MEMORIZACION

CALLEJON

SERVICIOS:
IMPRESION
CURSO EN PANTALLA DE FRENTE O REGRESO.

LOOP

BUSQUEDA EN MEMORIA

OTRO

DECISION AL AZAR

FIN

EXPLICACION DE LOS PROCESOS:

OLVIDO

ES UNA RUTINA DE SERVICIO QUE NOS PERMITE, COMO SU NOMBRE LO INDICA, OLVIDAR LAS RUTAS QUE SE HAYEN EN LA MEMORIA DEL SIMULADOR, ESTO ES MUY NECESARIO CUANDO SE CAMBIA DE LABERINTO, O BIEN SE DESEA HACER EL PROCESO MAS RAPIDO.

RECORDAR

ESTE PROCESO ES YA PARTE FUNDAMENTAL EN EL SISTEMA, PUES LO QUE HACE ES SACAR LA INFORMACION QUE ESTE GUARDADA EN LA MEMORIA DE LA MAQUINA, Y PREPERARLA PARA SU MAS FACIL MANIPULACION. SI EN LA RUTINA DE OLVIDO HAN SIDO BORRADAS TODAS LAS RUTAS, ESTA RUTINA NO SERA EJECUTADA POR EL SIMULADOR.

INTERFASE

ESTA ES LA RUTINA DE COMUNICACION DEL SIMULADOR CON EL MUNDO EXTERIOR. EN ELLA SE LLEVARA A CABO LA ENTRADA Y LA SALIDA DE LA INFORMACION NECESARIA.

CALLEJON

EN ESTA PARTE SE COMPROBARA SI EL SIMULADOR ESTA EN PRESENCIA DE UN CALLEJON SIN SALIDA Y SE EFECTUARA LA RUTINA DESCRITA EN EL ANALISIS.
ESTA RUTINA SE PUEDE DECIR QUE YA ES PARTE DE LA TOMA DE DECISION.
ES EVIDENTE LA RAZON POR LA CUAL SE COLOCO PRIMERO QUE LAS OTRAS TRES RUTINAS QUE LE SIGUEN, PUES LO PRIMORDIAL PARA EL SIMULADOR,ANTES DE PODER TOMAR UNA DECISION, ES SABER SI PUEDE SEGUIR ADELANTE.

LOOP

UNA VEZ QUE SABEMOS QUE NO ES UN CALLEJON PODEMOS REVISAR SI NO ES UN LOOP, Y SE PROCEDERA DE ACUERDO AL ANALISIS.

BUSQUEDA EN MEMORIA

UNA VEZ QUE SABEMOS QUE ESTAMOS EN UN PUNTO SIGNIFICATIVO, PROCEDEMOS A BUSCAR EN MEMORIA SEGUN AL ANALISIS.

DECISION AL AZAR

ESTA ES LA DECISION DESCRITA EN EL PROCESO SIMPLE, Y SOLO SE TOMA EN CASO DE NO HABERLO HECHO YA EN LA RUTINA ANTERIOR.

SALIENDO DE ESTE PROCESO SE REGRESA A LA INTERFASE, DONDE SE EJECUTARA LA DECISION TOMADA,Y SE TOMARA EL ESTADO DEL SIGUIENTE PUNTO.

HASTA AQUI SE HA CREADO UN CIRCULO, ES DECIR QUE SE OBTUVO LA SOLUCION DEL LABERINTO POR MEDIO DE UN PROCESO ITERATIVO, EL CUAL ESTA PERFECTAMENTE DELIMITADO.

AL TERMINAR DE RESOLVER EL LABERINTO, LO CUAL SE INDICARA EN LA RUTINA DE INTERFASE DE ALGUNA MANERA, EL SIMULADOR NOS LLEVARA A LAS SIGUIENTES RUTINAS.

MEMORIZACION

ESTE PROCESO AUTOMATICAMENTE GUARDARA LA RUTA ENCONTRADA.

SERVICIOS

ESTA RUTINA NOS PERMITIRA IMPRIMIR CUALQUIERA DE LAS RUTAS QUE ESTEN EN LA MEMORIA, O BIEN NOS PERMITIRA HACER TODO EL RECORRIDO DE ESTAS RUTAN EN LA PANTALLA, YA SEA DE IDA O DE REGRESO.

A CONTINUACION SE AGREGAN LOS DIAGRAMAS DE FLUJO DETALLADOS:

P R O C E S O

1 DIMENSIONA VARIABLES

2 INCIALIZA VARIABLES
3 LLAMA A LA RUTINA DE OLVIDO
LLAMA LA RUTINA DE RECUERDO
4 ASIGNA EL PUNTO 1

5 LLAMA LA RUTINA DE INTERFASE

6 SI CONTADOR=1 Y FIN SALTA A 12

7 SI CONTADOR=1 SALTA A 15

8 LLAMA POS_ORIMET

9 LIMPIA EL SIGUIENTE VECTOR

10 SI NO ES EL FIN SALTA 15

11 LLAMA MEMORIZA

12 LLAMA SERVICIO

13 SI DESEA HACER OTRO SALTA A 2

14 FIN DEL SISTEMA

15 SI NO (ES CONTADOR=1 Y CALLEJON), SALTA AL 18

16 CAMBIA LA ORIENTACION INICIAL,SALTA A 5

17 SALTA AL 15

18 SI ES CALLEJON LLAMA RUTINA DE CALLEJON

19 SI CONTADOR=1 SALTA 15

20 NO_LOOP = 0

21 LLAMA A LA RUTINA DE LOOP,

22 SI NO_LOOP <> 0 SALTA A 15

23 SI EXISTEN OTRAS RUTAS LLAMA LA RUTINA DE BUSCA

24 SI NO HA HABIDO DECISION LLAMA LA RUTINA DE DECISION

25 LLAMA RUTINA DE DIRECCION

26 INCREMENTA EL CONTADOR E INICIALIZA EL SIGUIENTE PUNTO

I N T E R F A S E

1 DIMENSIONA E INICIALIZA LAS VARIABLES

2 PONE LA PANTALLA CORRESPONDIENTE A LA ORIENTACION DEL VECTOR.

3 SI EL TIPO DE OPERACION ES 1 o 3, PONE EN LA PANTALLA LA DISTANCIA Y ES STATUS_2 (PREVIAMENTE FORMATEA EL STATUS, PARA QUE SIEMPRE CORRESPONDAN LOS LADOS)

4 SI EL TIPO DE OPERACION ES 2 o 3, LEE DE LA PANTALLA LA DISTANCIA, EL STATUS, Y LOS ASIGNA AL VECTOR, PREVIAMENTE FORMATEADOS.

5 FIN DE LA RUTINA

L C O P

1 REVISA QUE EL SIMULADOR ESTE EN UN LOOP

2 SI NO LO ESTA ENTONCES VA AL PASO SIETE

3 INVESTIGA EL GIRO DEL LOOP

4 CIERRA TODAS LAS ENTRADAS NO-UNICAS QUE CORRESPONDAN AL SENTIDO DEL GIRO DEL LOOP, ASI COMO TODAS LAS PARTES FRONTALES QUE DE ALGUNA MANERA QUEDEN APUNTANDO HACIA LA PARTE DE ADENTRO DEL LOOP CUANDO SE DEN VUELTAS A LA IZQUIERDA O A LA DERECHA, ESTO SE HACE EN TODOS LOS PUNTOS INTERMEDIOS ENTRE EL PASO INICIAL Y FINAL DEL LOOP.

5 SE INVESTIGA SI DESPUES DE CERRAR LAS PUERTAS AUN ES UN LOOP ABIERTO

6 SE PROCEDE DE ACUERDO AL ANALISIS SEGUN EL ESTADO EN QUE SE TENGAN LAS INDICACIONES DE GIRO, ORIENTACION, DIRECCION, VUELTA.

7 FIN DE LA RUTINA

O L V I D O

1 DIMENSIONA E INICIALIZA VARIABLES

2 POME LA PANTALLA DE OLVIDO

3 PREGUNTA SI SE DESEAN PURGAR LOS ARCHIVOS

4 SI LA RESPUESTA ES NO SALTA A 6

5 PURGA Y RECREA LOS ARCHIVOS ASOCIADOS AL SISTEMA

6 FIN DE LA RUTINA

M E M O R I Z A

1 DIMENSIONA E INICIALIZA VARIABLES

2 ABRIR ARCHIVOS

3 GUIA=GUIA+1

4 SI GUIA=11 ENTONCES, PURGA LA RUTA 10, GUIA=10

5 GUARDA LOS PUNTOS DE LA RUTA QUE NO ESTEN CONTENIDOS EN NINGUNA OTRA RUTA ANTERIOR.

6 EN CASO DE EXISTIR CONEXION CON ALGUNA RUTA, PONE EL NUMERO DE RUTA Y EL NUMERO DE PUNTO DE LA RUTA CON LA CUAL SE CONECTA, EN EL ULTIMO PUNTO.

7 CERRAR ARCHIVOS

8 FIN DE LA RUTINA

C A L L E J O N

```
1   SI STATUS_2 () "000" FIN DE LA RUTINA
2   SONIDO DE CAMPANA
3   DISPLAY " REGRESO DE (DISTANCIA) METROS"
4   BORRAR EL VECTOR
5   DECREMENTA EL CONTADOR
6   CIERRA LE ENTRADA QUE PERMITIO EL ACCESO A ESE CALLEJON
7   INCREMENTA EL SIMULADOR
8   LLAMA A LA RUTINA DE INTERFASE
9   RETARDO DE TIEMPO PARA PERMITIR AL USUARIO LA LECTURA
10  SALTA A 1
```

E L I S C O

1 INCREMENTA CONT_MEM

2 BUSCA USANDO TANTOS PUNTOS IGUALES COMO EL VALOR DE CONT_MEM

3 SI NO ENCUENTRA ALGUN SECTOR IGUAL EN NINGUNA RUTINA, SALTA A 5

4 ANOTA EL NUMERO DE RUTA, LA POSICION EN ELLA, LA POSICION EN EL VECTOR EN CUESTION, Y EL CONT_MEM. FIN DE LA RUTINA

5 DECREMENTA EL CONT_MEM

6 SI CONT_MEM=0 ENTONCES BORRA LA BASURA DE LAS VARIABLES. FIN DE LA RUTINA

7 SALTA A 2

B E G O R D A R

1 DIMENSIONAR E INICIALIZAR VARIABLES

2 ABRIR ARCHIVOS

3 LEE EL ARCHIVO GUIA Y OBTIENE EL NUMERO DE RUTAS.

4 GUARDA LAS GUIAS DE LAS RUTAS EN EL VECTOR VEC_GUIA

5 LEE RUTA POR RUTA EL NUMERO DE REGISTROS QUE ESTE EN ARCHIVO

6 EN CASO DE QUE EL VECTOR GUIA INDIQUE QUE HAY MAS NUMERO DE PUNTOS QUE REGISTROS EN EL ARCHIVO, ENTONCES OBTENER LOS PUNTOS RESTANTES DEL VECTOR CON EL CUAL ESTE CONECTADA LA RUTA QUE SE ESTE PROCESANDO EN ESE MOMENTO, ORIENTANDO CADA PUNTO DE ACUERDO A ESTA RUTA, Y NO A LA RUTA QUE ESTA EN MEMORIA.

7 CERRAR LOS ARCHIVOS

8 FIN DE LA RUTINA

S E R V I C I O

```
1   DIMENSIONA E INICIALIZA VARIABLES
2   PONE LA PANTALLA DE SERVICIO
3   PIDE LA OPCION DESEADA
4   SI NO ES EL FINAL SALTA 6
5   PREGUNTA SI SE DESEA HACER OTRA RUTA, Y GUARDA LA RESPUESTA EN UNA
    VARIABLE. FIN DE LA RUTINA
6   ASIGNA UN VECTOR AUXILIAR CON LA RUTA DESEADA.
7   SI SE DESEA IMPRESION, LLAMA A LA RUTINA DE IMPRESION
8   SI SE DESEA REPETICION, LLAMA A LA RUTINA DE FRENTE
9   SI SE DESEA REGRESO, LLAMA A LA RUTINA DE REGRESO
10  PONE LA PANTALLA DE SERVICIO
```

7.- CREACION DEL SISTEMA QUE USARA EL SIMULADOR

- RESTRICCIONES DE LA COMPUTADORA ELEGIDA.

LA COMPUTADORA QUE SE USO PARA IMPLEMENTAR EL SISTEMA ES UNA MINI-COMPUTADORA HEWLETT-PACKARD 300.

ESTA COMPUTADORA TIENE UNA CAPACIDAD DE MEMORIA PRINCIPAL, O DE PROCESO DE 1 MEGABYTE. Y MEMORIA EN DISCO RIGIDO DE 12 MEGABYTES.

EL COMPILADOR PRINCIPAL DE ESTA MAQUINA ES DE LENGUAJE BASIC, Y OFRECE GRANDES FACILIDADES EN EL MANEJO DE SUBSTRINGS, PORQUE ESTE BASIC ESTA ESPECIALMENTE ORIENTADO A LOS NEGOCIOS, POR LO QUE SE LE LLAMA BUSINESS BASIC.

LA RAZON POR LA CUAL SE USO, ES PORQUE SE TENIA ACCESO ILIMITADO AL USO DE ESTE COMPUTADOR, Y AUNQUE DE ANTEMANO SE SABIA QUE EL BASIC NO ES UN LENGUAJE MUY PODEROSO, COMO EL PROBLEMA NO ERA SUMAMENTE COMPLICADO, SE PUDO HACER EL PROGRAMA EN EL.

EL PRINCIPAL DEFECTO CON QUE CUENTA ESTA MAQUINA ES QUE ES DEMASIADO LENTA, LO CUAL SE DEBE A QUE TIENE UN SISTEMA OPERATIVO QUE SIEMPRE ESTA TRATANDO DE REVISAR CUANTO COMANDO O INSTRUCCION SE LE DE, PARA EN CASO DE EXISTIR UN ERROR, TENER SIEMPRE LA POSIBILIDAD DE MANDAR UN MENSAJE AL OPERARIO. POR ESTA RAZON SE PRESTA MUCHO PARA UNA FACIL DEPURACION DE LOS PROGRAMAS, PERO EL RESULTADO, ES QUE EL TIEMPO EN QUE ESTOS CORREN ES DEMASIADO, COMPARADO CON OTROS PROCESADORES.

DEBIDO A ESTE PROBLEMA, SE TUVO QUE PONER CIERTAS RESTRICCIONES AL PROGRAMA DEL SIMULADOR.

PARA EMPEZAR SE SABIA QUE SE PODIA LLEGAR A MANEJAR UNA GRAN CANTIDAD DE INFORMACION. ESTO NO ERA GRAN PROBLEMA CONSIDERANDO LA CAPACIDAD DEL DISCO CON QUE CUENTA LA MAQUINA.

DE ALGUNA U OTRA FORMA, LA INFORMACION GENERADA EN CADA RESOLUCION DE UN LABERINTO TENDRIA QUE SER GUARDADA PERMANENTEMENTE EN EL DISCO RIGIDO, PARA QUE NO IMPORTANDO EN QUE MOMENTO SE DECIDIERA USAR EL SIMULADOR ESTE SIEMPRE TUVIERA LA INFORMACION NECESARIA PARA AYUDARSE A SALIR.

EL DILEMA SE PRESENTA EN EL MOMENTO DE DECIDIR COMO MANEJAR LA INFORMACION, YA QUE AL ESTAR PROCESANDO UNA NUEVA RUTA EXISTEN DOS OPCIONES POSIBLES:

1.- MANEJARLA DIRECTAMENTE EN LOS ARCHIVOS (DISCO RIGIDO)

LA UNICA VENTAJA VISIBLE ES QUE CUMPLE CON UNA DE LAS CARACTERISTICAS QUE SE DIO EN EL PLANTEAMIENTO DEL PROBLEMA; Y ESTA ES QUE EL LIMITE DE PUNTOS O DE RUTAS QUE PODRIA TENER EL SIMULADOR, ESTARIAN DADOS SOLAMENTE POR LA CAPACIDAD DE LA MAQUINA. Y SI MANEJAMOS ADECUADAMENTE LOS ARCHIVOS, PODRIAMOS LLEGAR A TENERLOS TAN GRANDES COMO LO QUE LA MAQUINA NOS PERMITIERA.

LA DESVENTAJA QUE PRESENTA ESTA FORMA, ES QUE EL TIEMPO DE ACCESO

AL DISCO, ES SUMAMENTE LENTO. ESTO NO OCASIONARIA PROBLEMAS, SI SOLO SE TRATARA DE RESOLVER EL LABERINTO SIN TENER EN CUENTA LAS RUTAS QUE SE HUBIESEN MEMORIZADO, PUESTO QUE POR CADA PUNTO QUE SE TRATARA DE REVISAR, SE TENDRIA QUE DAR SOLO UNA BAJA O UNA ALTA, LO CUAL SE HACE MUY RAPIDAMENTE, SI HABLAMOS EN TERMINOS HUMANOS.

PERO EL OBJETIVO DEL SIMULADOR ES PRECISAMENTE DEMOSTRAR QUE DE ALGUNA MANERA LA MAQUINA HA TENIDO UN APRENDIZAJE ARTIFICIAL, Y QUE PUEDE USAR LA INFORMACION QUE HA ADQUIRIDO, PARA AYUDARSE A SI MISMA.

ESTO QUIERE DECIR, QUE EL SIMULADOR DEBE TENER PRESENTE TODA LA INFORMACION QUE ESTA EN SU MEMORIA ANTES DE PODER TOMAR CUALQUIER DECISION.

COMO A CADA PASO DEBE DE TOMAR UNA DECISION, IMAGINEMOS EL TIEMPO QUE TARDARIA EN EL MOMENTO QUE TUVIERA 2 O 3 RUTAS GUARDADAS EN SU MEMORIA. ESTO NO SERIA PROBLEMATICO PARA EL SIMULADOR, PERO PARA LA PERSONA QUE ESTE USANDOLO, SERIA SUMAMENTE TEDIOSO. POR EJEMPLO, IMAGINEMOS UN LABERINTO EN EL CUAL YA SE HAN ENCONTRADO TRES RUTAS. SUPONGAMOS QUE CADA RUTA FUE EN PROMEDIO DE 50 PASOS. SI EN LA NUEVA RUTA EL SIMULADOR VA YA EN EL PASO 15, Y PIENSA QUE YA HA ENCONTRADO 9 PUNTOS DE UNA RUTA, Y DE REPENTE AL HACER LA REVISION 10, NO ENCUENTRA YA NINGUN PUNTO QUE COINCIDA, SIMPLE Y SENCILLAMENTE PORQUE LOS 9 PUNTOS QUE HABIA ENCONTRADO CORRESPONDIAN A UNA PARTE NUEVA QUE POR CASUALIDAD ERA IDENTICA EN A UNA PARTE DE UNA RUTA CONOCIDA. ENTONCES SE SUPONE QUE EMPEZARA A REVISAR CADA VEZ MENOS PUNTOS, Y SI POR MALA SUERTE NO ENCONTRASE NINGUNO, EN ESE MOMENTO EL SIMULADOR HABRIA LEIDO 10 VECES EL ARCHIVO, ES DECIR 1500 LECTURAS, Y ESTO TAN SOLO PARA DECIDIR SI SE DA VUELTA A LA IZQUIERDA, DERECHA, O FRENTE. SI SE TIENE EN CUENTA QUE APARTE SE TIENE QUE FORMATEAR LA INFORMACION PARA PODER COMPARARLA, EL TIEMPO QUE ESTO TARDARIA SERIA DEMASIADO.

2.- PROCESAR LA INFORMACION DIRECTAMENTE EN LA MEMORIA PRINCIPAL.

ESTA OPCION ELIMINA LA DESVENTAJA QUE PRESENTA EL MANEJO DE INFORMACION DIRECTAMENTE EN LOS ARCHIVOS, PUESTO QUE LO QUE SE HACE, ES LEER TODA LA INFORMACION EN EL MOMENTO EN QUE SE EMPIEZA A CORRER EL PROGRAMA, LA GUARDA EN UNAS VARIABLES AUXILIARES, LAS CUALES SE USARAN MAS TARDE PARA LA BUSQUEDA EN MEMORIA. DE ESTA MANERA, SOLAMENTE SE LEEN LOS ARCHIVOS UNA SOLA VEZ EN TODO EL PROGRAMA.

COMO COMPARACION DE TIEMPOS, SE HIZO LA ASIGNACION EN MEMORIA PRINCIPAL DE 50,000 VARIABLES. EL TIEMPO QUE TARDO EL COMPUTADOR FUE DE 7 SEGUNDOS 85 CENTESIMAS. POR OTRA PARTE SE HICIERON TAMBIEN 50,000 LECTURAS A UN ARCHIVO CONVENCIONAL, OBTENIENDOSE UNN TIEMPO DE 5 MINUTOS Y 30 SEGUNDOS. EL TIEMPO DE ASIGNACION EN MEMORIA PRINCIPAL FUE APROXIMADAMENTE EL 2.5% DEL TIEMPO DE LECTURA A EL ARCHIVO. CON ESTO SE COMPRUEBA QUE LA GANACIA ES DEFINITIVA SI SE USA LA SEGUNDA OPCION.

SE PENSO QUE LA MEJOR MANERA DE MANEJAR LA INFORMACION QUE SE OBTUVIERA DE LOS ARCHIVOS, ERA GUARDARLA EN UNA MATRIZ, CUYO NUMERO DE RENGLONES FUERA IGUAL AL NUMERO DE RUTAS RECORRIDAS, Y EL NUMERO DE COLUMNAS CORRESPONDIENTE A NUMERO DE PUNTOS QUE TUVIERA LA RUTA MAS LARGA.

EL PROBLEMA QUE SE PRESENTO ES QUE EL PROCESADOR TIENE UN LIMITE EN CUANTO AL NUMERO DE ELEMENTOS QUE ES CAPAZ DE DIRECCIONAR EN SU MEMORIA PRINCIPAL PARA UNA SOLA VARIABLE. ESTE NUMERO ES IGUAL A 65,536 LOCALIDADES, ES DECIR LO QUE PUEDEN DIRECCIONAR 16 BITS.

LA INFORMACION MINIMA QUE PUEDEN TENER EN ESTE CADA ELEMENTO DE LA MATRIZ PARA PODER TRABAJAR EL SIMULADOR ES DE 28 (LUEGO SE DEFINIRAN). SI DIVIDIMOS 65,536 ENTRE 28 NOS DA UNA MATRIZ DE 2340 ELEMENTOS APROXIMADAMENTE. ESTO ES QUE NUESTRA MATRIZ PUEDE TENER POR EJEMPLO 20 RUTAS DE 100 ELEMENTOS CADA UNA.

ESTA MAGNITUD DE 2340 ELEMENTOS ES DEMASIADO CHICA, POR LO CUAL SE DECIDIO ATACAR EL PROBLEMA DESDE OTRA MANERA.

SE PENSO DEFINIR VECTORES QUE TUVIERAN 2000 ELEMENTOS CADA UNO, Y VER CUAL ERA MAYOR CANTIDAD DE VECTORES DE ESTA CAPACIDAD QUE LA MAQUINA ERA CAPAZ DE DIRECCIONAR SIN PROBLEMAS. DESPUES DE MUCHAS PRUEBAS SE LLEGO A LA CONCLUSION DE QUE UN NUMERO OPTIMO ERA EL DE 10 VECTORES.

POR TANTO LA PRIMERA RESTRICCION QUE TENDRA EL SIMULADOR SERA LA SIGUIENTE:

- SERA CAPAZ DE TENER GUARDADAS EN SU MEMORIA, HASTA 10 RUTAS DE 2000 ELEMENTOS CADA UNA.

- A LA VEZ ESTARA PROCESANDO UNA RUTA QUE SERA TAMBIEN DE 2000 ELEMENTOS.

VENTAJAS:

- REDUCCION DEL TIEMPO DE RESPUESTA DEL SIMULADOR DE 97.5%

DESVENTAJAS:

- REDUCCION DE LAS DIMENSIONES DEL LABERINTO.

ESTA ULTIMA DESVENTAJA NO ES MUY RELEVANTE SI SE CONSIDERA QUE UN LABERINTO QUE CON MAS DE 2000 POSIBLES NO ES MUY COMUN EN LA VIDA REAL.

EL SEGUNDO PROBLEMA QUE PRESENTA ESTA MAQUINA ES QUE NO TIENE UNA GRAN CAPACIDAD EN EL MANEJO DE PANTALLA, POR LO CUAL NO SE PUDO HACER QUE EL LABERINTO APARECIERA EN LA PANTALLA CONFORME EL SIMULADOR LO FUERA DESCUBRIENDO. PERO LO QUE SE HIZO FUE PONER UNA FIGURA EN LA CUAL SE RECONOCE PERFECTAMETE CUAL ES LA PARTE FRONTAL, Y LOS LADOS DEL SIMULADOR, ASI COMO SE VEN LOS SENSORES DE QUE SE HABLO ANTERIORMENTE.

ESTA FIGURA PODRA ESTAR ORIENTADA HACIA CUALQUIERA DE LOS EJES CARTESIANOS, Y MEDIANTE LA ORIENTACION QUE TOME EN UN MOMENTO DETERMINADO, NOS INDICARA HACIA DONDE DIO LA VUELTA Y EMPEZO A CAMINAR.

EN ESTA MISMA PANTALLA SE LE PODRAN INTRODUCIR LOS DATOS CORRESPON-

DIENTES A DISTANCIA RECORRIDA Y STATUS, Y LA CAPURA DE LOS MISMOS SERA SUMAMENTE FACIL.

--0--

D E S C R I P C I O N D E A R C H I V O S

ARCHIVO GUIA

NOMBRE: TESGUIA(JJTESIS)
LLAVE : NUMERO DE RUTA
TIPO DE LLAVE : UNICA
FORMATO : 4 CARACTERES

NUMERO DE RUTA 9(4)	[1 ,4]
NUMERO DE REGISTROS 9(10) NUMERO DE REGISTROS GUARDADOS EN EL ARCHIVO.	[5 ,14]
NUMERO DE PUNTOS 9(10) NUMERO DE PUNTOS QUE TIENE LA RUTA	[15,24]
FILLER ESPACIO VACIO	[25,50]

ARCHIVO DE RUTAS

NOMBRE: TESRUTA(JJTESIS)
LLAVE : NUMERO DE RUTA
TIPO DE LLAVE : NO UNICA
FORMATO : 4 CARACTERES

NUMERO DE RUTA 9(4)	[1 ,4]
DISTANCIA 9(5)	[5 ,9]
POSICION EN EL EJE X 9(5)	[10,14]
SIGNO (ASOCIADO A LA POS_X)	[15,15]

U N I V E R S I D A D L A S A L L E

DIRECCION GENERAL DE SERVICIOS ESCOLARES

* FAVOR DE LLENAR POR TRIPLICADO CON LETRA DEL MOLDE
* ENTREGAR DOS EJEMPLARES DE LA TESIS EN LA BIBLIOTECA CENTRAL-UNAM
* EXIGIR QUE LE SELLEN LAS DOS COPIAS

NOMBRE DEL ALUMNO	No. CTA. UNAM
TAMAMANTES DEL TORO JUAN JOSE	196788

NOMBRE DE LA TESIS O SEMINARIO
INTRODUCCION AL CONCEPTO DE INTELIGENCIA ARTIFICIAL MEDIANTE LA IMITACION DE UN PROCESO DE APRENDIZAJE

ESCUELA O UNIVERSIDAD	CARRERA
UNIVERSIDAD LA SALLE	ING. ELECTRONICA

FECHA	DIA	MES	AÑO
	24	MAYO	1983

EXCUSE DE RECIBO, SELLO Y FIRMA DE LA BIBLIOTECA

* ORIGINAL BIBLIOTECA UNAM.

X(1)
(+/-)

POSICION EN EL EJE Y [16,20]
9(5)

SIGNO (ASOCIADO A LA POS_Y) [21,21]
X(1)
(+/-)

STATUS 1 [22,24]
9(3)
(0/1)----- D1, F1, I1

STATUS 2 [25,27]
9(3)
(0/1)----- D2, F2, I2

ORIENTACION [28,29]
X(2)
+X, -X, +Y, -Y

VUELTA [30,30]
X(1)
F = FRENTE
D = DERECHA
I = IZQUIERDA

DIRECCION [31,32]
X(2)
+X, -X, +Y, -Y

D A T O S D E C O N E X I O N

RUTA A CONECTAR [33,36]
9(4)

NUMERO DE REGISTRO DE CONEXION EN LA RUTA [37,46]
9(10)

INDICADOR DE FIN [47,47]
X(1)

```
"F"  0  " "

FILLER                                          [48,70]
   ESPACIO VACIO
```

DESCRIPCION DE VECTORES

DISTANCIA	(1 ,5)
POSICION EN X	(6 ,10)
SIGNO_X	(11,11)
POSICION EN Y	(12,16)
SIGNO_Y	(17,17)
D1	(18,18)
F1	(19,19)
I1	(20,20)
D2	(21,21)
F2	(22,22)
I2	(23,23)
ORIENTACION	(24,25)
VUELTA	(26,26)
DIRECCION	(27,28)

LA INFORMACION QUE MANEJEN LOS VECTORES DEBERA ESTAR DE ESTA MANERA, Y ES POR ESO QUE EN LOS PROGRAMAS SE DIMENSIONAN VECTORES DE 28 CAMPOS ALFANUMERICOS.

-- 0 --

EL PROGRAMA TIENE LA POSIBILIDAD DE CORRER TENIENDO EN CUENTA LA INFORMACION GENERADA EN RUTAS ANTERIORES O NO.

LA FORMA DE HACERLO ES LA SIGUIENTE.

A) SI SE DESEA QUE EL SISTEMA TENGA EN CUENTA LAS RUTAS ANTERIORES SE DEBE CREAR UN DOMINIO CUYO NOMBRE ES "JITCSIS". PUES LOS DOS ARCHIVOS QUE USA EL SIMULADOR SERAN CREADOS POR EL EN ESE DOMINIO.

B) SI NO SE DESEA QUE EXISTA APRENDIZAJE, SIMPLEMENTE NO SE CREA ESE DOMINIO, Y EL SIMULADOR, NO SOLO NO SERA CAPAZ DE RECORDAR LAS RUTAS, ANTERIORES, SINO QUE NO PODRA NI SIQUIERA GUARDAR LA RUTA ENCONTRADA.

8.- CODIFICACION

EN ESTA PARTE ESTA ESCRITO TODO EL SISTEMA QUE INTEGRA AL SIMULADOR, ESTA DIVIDIDO EN MODULOS, SEGUN LA SEGMENTACION QUE SE PENSO SERIA LA MAS ADECUADA PARA FACILITAR LA TAREA DEL COMPILADOR.

ALGUNAS LINEAS ESTAN ESCRITAS EN DOS RENGLONES. EN REALIDAD EL COMPILADOR BASIC/300 TAN SOLO ACEPTA INSTRUCCIONES ESCRITAS EN UNA SOLA LINEA, PERO CON EL OBJETO DE QUE EL PROGRAMA CUPIERA EN HOJAS CARTA, SE DECIDIO DIVIDIR LAS INSTRUCCIONES QUE FUERAN DEMASIADO LARGAS EN DOS LINEAS, CON LA SALVEDAD DE QUE SI SE DESEA IMPLEMENTAR ESTE PROGRAMA, SE ESCRIBAN ESTAS INSTRUCCIONES EN UNA SOLA.

EL PROGRAMA ASI COMO ESTA SOLO PUEDE SER IMPLEMENTADO EN UNA COMPUTADORA HP-300. SIN EMBARGO, HACIENDO LAS MODIFICACIONES NECESARIAS PUEDE ADAPTARSE A CUALQUIER COMPUTADORA QUE TENGA BASIC, Y QUE SEA CAPAZ DE MANEJAR SUBSTRINGS Y VECTORES ALFANUMERICOS CON RELATIVA FACILIDAD.

EL PRIMER MODULO SOLO SIRVE PARA QUE EL SIMULADOR SEPA CUANTOS MODULOS TIENE QUE COMPILAR.

M O D U L O D E I N I C I O (MAIN)

```
10 $TITLE="$$$ SIMULADOR EJEMPLO DE UN PROCESO DE APRENDIZAJE $$$"
20 !
30 !
40 !
50 $SUBTITLE="$$$ MODULO DE CONTROL DEL SIMULADOR $$$       (PROCESO)"
60 $PAGE
70 $INCLUDE PROCESO
80 !
90 !
100 !
110 $SUBTITLE="$$$ MODULO DE INTERFASE CON EL USUARIO $$$   (INTERFAS)"
120 $PAGE
130 $INCLUDE INTERFAS
140 !
150 !
160 !
170 $SUBTITLE="$$$ MODULO DE RUTINAS AUXILIARES $$$         (RUTINAS)"
180 $PAGE
190 $INCLUDE RUTINAS
200 !
210 !
220 !
```

```
$SUBTITLE="*** MODULO DE REVISION DE LOOP ***                (LOOP)"
$PAGE
$INCLUDE LOOP
!
!
!
$SUBTITLE="*** MODULO DE OLVIDO DE RUTAS ***               (OLVIDO)"
$PAGE
$INCLUDE OLVIDO
!
!
!
$SUBTITLE="*** MODULO DE MEMORIZACION DE RUTAS ***       (MEMORIZA)"
$PAGE
$INCLUDE MEMORIZA
!
!
!
$SUBTITLE="*** MODULO DE RECUERDO ***                    (RECORDAR)"
$PAGE
$INCLUDE RECORDAR
!
!
!
$SUBTITLE="*** MODULO DE BUSQUEDA EN MEMORIA ***            (BUSCA)"
$PAGE
$INCLUDE BUSCA
!
!
!
$SUBTITLE="*** MODULO DE RUTINAS DE SERVICIO ***         (SERVICIO)"
$PAGE
$INCLUDE SERVICIO
```

```
$CONTROL SEGMENT PROCESO
$CONTROL MAP
!
!
!
!
!        SIMULADOR CREADO POR JUAN JOSE TALAMANTES DEL TORO
!        MEXICO D.F. A 6 DE NOVIEMBRE DE 1982
!
!
!
!*****************************************************************
!*****           DIMENSIONA VARIABLES                        *****
!*****************************************************************
```

```
1140    !
1150    INTEGER  Term_id
1160    INTEGER  Row
1170    INTEGER  Col
1180    INTEGER  Status(1:2)
1190    INTEGER Tiop
1200    INTEGER I,J,K
1210    INTEGER Arquia_id
1220    INTEGER Arrsta_id
1230    INTEGER Caja
1240    INTEGER No_Inop
1250    INTEGER Contador
1260    INTEGER Vec_men
1270    INTEGER Pos_men
1280    INTEGER Pos_vec
1290    INTEGER Cont_men
1300    !
1310    DIM Line$[220]
1320    DIM Input$[11]
1330    DIM Perpon$[10]
1340    DIM Term$[8]
1350    DIM Error$[220]
1360    DIM Vector$(1:2000)[20]
1370    DIM Vec_guia$(1:10)[50]
1380    !
1390    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1400    !!!!!!                  ABRE LA CONSOLA                          !!!!!
1410    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1420    !
1430    ICALL DGETPARMS(Parms$,ALEN(Parms$))
1440    Term$=TRIM$(Parms$[1,8])
1450    IF LEN(Term$)=0 THEN Term$=".CONSOLE"
1460    ASSIGN @1 TO Term$
1470    ICALL FOFILEID(@1), Term_id
1480    !
1490    ICALL THDSPCNTL(Term_id,21,3,21,Status(*)) !Posiciona el cursor Col=3 Row=21
1500    ICALL THDSPCNTL(Term_id,8,,,Status(*)) !Limpia el renglon y abajo
1510    PRINT @1;    "                       INICIALIZACION DE VARIABLES
1520    !
1530    DIM V1$(1:2000)[20]
1540    DIM V2$(1:2000)[20]
1550    DIM V3$(1:2000)[20]
1560    DIM V4$(1:2000)[20]
1570    DIM V5$(1:2000)[20]
1580    DIM V6$(1:2000)[20]
1590    DIM V7$(1:2000)[20]
1600    DIM V8$(1:2000)[20]
1610    DIM V9$(1:2000)[20]
1620    DIM V0$(1:2000)[20]
1630    !
1640    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1650    !!!!!!              COMIENZA UNA NUEVA RUTA                      !!!!!
1660    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1670    !
```

```
1680 Nueva_ruta:
1690    !
1700    Vec_mem=0
1710    Pos_mem=0
1720    Pos_vec=0
1730    Guia=0
1740    Cont_mem=0
1750    FOR I=1 TO 2000
1760    Vector$(I)[1,28]=""
1770    NEXT I
1780    !
1790    !**************************************************************
1800    !*****   LLAMA A LA RUTINA DE OLVIDO DE RUTAS ANTERIORES   *****
1810    !**************************************************************
1820    !
1830    CALL Olvido(@1,Term_id)
1840    !
1850    !**************************************************************
1860    !*****  LLAMA A LA RUTINA DE RECUERDO DE RUTAS ANTERIORES  *****
1870    !**************************************************************
1880    !
1890    CALL Recordar(@1,V10(),V20(),V30(),V40(),V50(),V60(),V70(),
                  V80(),V90(),V00(),Vec_guia$(),Guia,Term_id)
1900    !
1910    !**************************************************************
1920    !*****             INICIALIZA EL PRIMER PUNTO              *****
1930    !**************************************************************
1940    !
1950    Contador=1
1960    Vector$(1)[1,17]="    0     0+    0+"
1970    Vector$(1)[24;2]="+X"
1980    Tiep=3
1990    !
2000    !**************************************************************
2010    !*****           SE LLAMA A LA RUTINA DE PANTALLA          *****
2020    !**************************************************************
2030    !
2040 Llama_pantalla:
2050    !
2060    CALL Interfase(@1,Vector$(Contador),Tiep,Term_id)
2070    IF Vector$(1)[27,28]="F1" THEN Llama_servicio
2080    IF Contador=1 THEN Checa_callejon
2090    !
2100    !**************************************************************
2110    !*****     LLAMA LA RUTINA DE ORIENTACION Y POSICION       *****
2120    !**************************************************************
2130    !
2140    CALL Pos_orient(Vector$(Contador-1),Vector$(Contador))
2150    Vector$(Contador+1)[1,28]=""
2160    !
2170    !**************************************************************
2180    !*  EN CASO DE REGRESAR CON LA INDICACION DE FIN EN EL VECTOR, *
2190    !*  LLAMA A LA RUTINA DE MEMORIZACION Y LA DE SERVICIO.        *
2200    !**************************************************************
```

```
2210    !
2220    IF Vector$(Contador)[27,28]="FI" THEN
2230    CALL Memoriza(01,Vector$(),Contador,Gaia,
                      Vec_mem,Pos_mem,Pos_vec,Term_id)
2240    !
2250 Llama_servicio:
2260    !
2270    CALL Servicio(01,Otra$,V1$(),V2$(),V3$(),V4$(),V5$(),V6$(),V7$(),V8$(),
                      V9$(),V10$(),Vector$(),Vec_gaia$(),Contador,Gaia,Term_id)
2280    !
2290    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2300    ! SI SE DESEA HACER UNA NUEVA RUTA, SE REGRESA A LA ETIQUETA DE
2310    ! NUEVA RUTA Y COMIENZA TODO EL PROCESO, DESDE LA INICIALIZACION
2320    ! DE LAS VARIABLES .
2330    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2340    !
2350    IF Otra$[1,1]="S" THEN Nueva_ruta
2360    GOTO Fin
2370    END IF
2380    !
2390    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2400    !!!!!! EN CASO DE SER NECESARIO LLAMA A LA RUTINA DE CALL- !!!!!
2410    !!!!!! JON. SI ES EL PRIMER PUNTO Y ESTA EN UN CALLEJON,   !!!!!
2420    !!!!!! SE DA UNA VUELTA DE 180 GRADOS E REINICIA EL PROCESO!!!!!
2430    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2440    !
2450 Checa_callejon:
2460    !
2470    IF Vector$(Contador)[21,23]="000" AND Contador=1 THEN
2480    IF Vector$(1)[24,1]="*" THEN
2490    Vector$(1)[24,1]="-"
2500    ELSE
2510    Vector$(1)[24,1]="*"
2520    END IF
2530    Ver_status$[1,1]=Vector$(1)[18,1]
2540    Vector$(1)[18,1]=Vector$(1)[28,1]
2550    Vector$(1)[28,1]=Ver_status$[1,1]
2560    GOTO Llama_pantalla
2570    END IF
2580    IF Vector$(Contador)[21,23]="000" THEN
2590    CALL Callejon(01,Vector$(),Contador,Cont_mem,Term_id)
2600    IF Contador=1 THEN Checa_callejon
2610    END IF
2620    No_loop=0
2630    CALL Loop(01,Vector$(),Contador,No_loop,Cont_mem,Term_id)
2640    IF No_loop()0 THEN Checa_callejon
2650    !
2660    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2670    !!!!!! EN CASO DE EXISTIR RUTAS ANTERIORES, LLAMA A LA RU- !!!!!
2680    !!!!!! TINA DE BUSQUEDA EN LA MEMORIA                     !!!!!
2690    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2700    !
2710    IF Gaia()0 THEN
2720    CALL Busco(01,V1$(),V2$(),V3$(),V4$(),V5$(),V6$(),V7$(),V8$(),V9$(),V10$(),
```

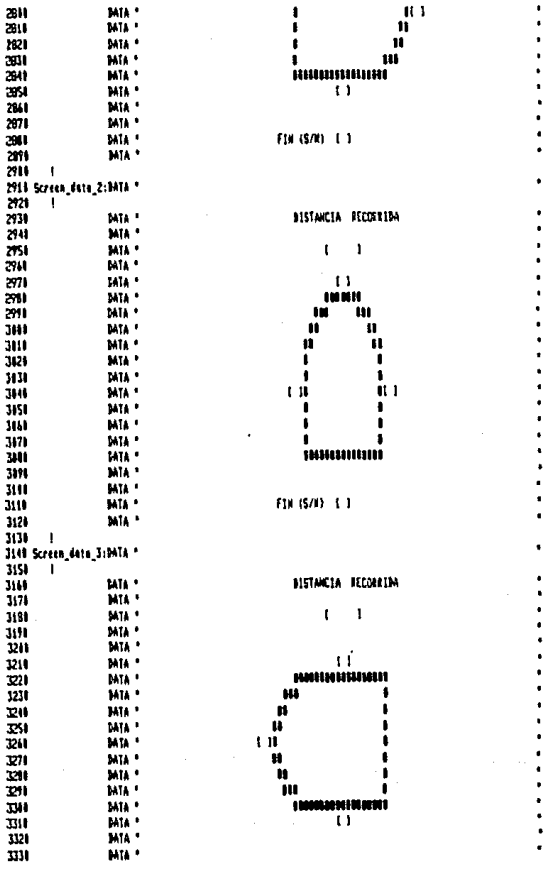
```
                    Vec_guia$(),Vector$(),Guia,Contador,Vec_mem,Pos_mem,Pos_vec,Cont_mem,Term_id)
2730    ENDIF
2740    !
2750    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2760    !!!!!! EN CASO DE NO ENCONTRAR NADA EN SU MEMORIA, TOMA LA !!!!!
2770    !!!!!! DECISION AL AZAR                                    !!!!!
2780    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2790    !
2800    IF Vector$(Contador)[26;1]=" " THEN CALL Decision_azar(Vector$(Contador))
2810    !
2820    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2830    !!!!!! LLAMA A LA RUTINA DE DIRECCION                      !!!!!
2840    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2850    !
2860    CALL Direccion(Vector$(Contador))
2870    !
2880    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2890    !!!!!! INCREMENTA EL CONTADOR, PREPARA EL SIGUIENTE PUNTO  !!!!!
2900    !!!!!!  CAR EL STATUS DEL SIGUIENTE PUNTO.                 !!!!!
2910    !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2920    !
2930    Contador=Contador+1
2940    Vector$(Contador)[24,25]=Vector$(Contador-1)[27,28]
2950    GOTO Llama_pantalla
2960    !
2970 Fin:
2980    !
2990    ASSIGN @1 TO *
3000    END
```

```
1000    $CONTROL SEGMENT INTERFAS
1010    $CONTROL MAP
1020    !
1030    !
1040    !
1050    !
1060    !
1070 SUB Interfase(S$,Bufpanta$,INTEGER Temp,Term_id)
1080    !
1090    !
1100    !
1110    !
1120    !
1130    INTEGER  Row
1140    INTEGER  Col
1150    INTEGER  K
1160    INTEGER  Status(1:2)
1170    DIM Line$(220)
```

```
1180    DIM Input$[13]
1190    DIM Patpan$[11]
1200    DIM Error$[220]
1210    DIM Or$[2]
1220    !
1230    !POWE LA PANTALLA CORRESPONDIENTE A LA ORIENTACION DEL VECTOR
1240    !
1250    GOSUB Inicio_limpio
1260    Or$[1,2]=Bafpantos[24,25]
1270    IF Or$="X" THEN RESTORE Screen_data_1
1280    IF Or$="Y" THEN RESTORE Screen_data_2
1290    IF Or$="-X" THEN RESTORE Screen_data_3
1300    IF Or$="-Y" THEN RESTORE Screen_data_4
1310    FOR   X=0 TO 20
1320    Line$[1,220] = ""
1330    READ Line$
1340    PRINT 01; Line$
1350    NEXT X
1360    !
1370 Opcion:
1380    !
1390    GOSUB Fin
1400    Patpan$[1,11]=""
1410    IF Tiep=3 OR Tiep=1 THEN
1420    !
1430    !ASIGNA LA DISTANCIA PARA POWERLA EN LA PANTALLA
1440    !
1450    Patpan$[3,7]=Bafpantos$[1,5]
1460    '
1470    IF Or$="+X" THEN Patpan$[8,3]=Bafpantos$[20,11]&Bafpantos$[19,11]&Bafpantos$[18,11
1480    '
1490    IF Or$="-X" THEN Patpan$[8,3]=Bafpantos$[18,11]&Bafpantos$[19,11]&Bafpantos$[20,11
1500    '
1510    IF Or$="+Y" THEN Patpan$[8,3]=Bafpantos$[19,11]&Bafpantos$[20,11]&Bafpantos$[18,11
1520    !
1530    IF Or$="-Y" THEN Patpan$[8,3]=Bafpantos$[18,11]&Bafpantos$[20,11]&Bafpantos$[19,11
1540    !
1550    END IF
1560
1570    Patpan$[11,11]="N"
1580    !CALL PUTA(Term_id,Patpan$,LEN(Patpan$),Status())
1590    GOSUB Faff
1600    Tow=3
1610    Col=33
1620    '
1630     REVISA SI TAMBIEN HAY QUE LEER, Y SI NO SE VA AL FINAL
1640    !
1650    IF Tiep()3 AND Tiep()2 THEN Fin
1660    GOSUB Limpio_21
1670    PRINT 01;     "          Teclee la las datos del siguiente punto y luego 'ENTER'      "
1680    !
1690 Gets_term:
1700    !
1710    GOSUB Fin
```

```
1720    GOSUB Fpos
1730    Input$[1,13]=""
1740    ICALL GETN(Term_id,Input$,LEN(Input$),Status())
1750    GOSUB Foff
1760    Error$=""
1770    !
1780    !REVISA QUE LA DISTANCIA SEA CAMPO NUMERICO
1790    !
1800    IF TRIM$(Input$[1,5])="" THEN
1810    Error$="                    SE ESPERAN DATOS EN ESTE CAMPO
1820    GOTO Disp_error
1830    ENDIF
1840    FOR I=1 TO 5
1850    IF POS(" 1234567890",Input$[I;1])=0 THEN
1860    Error$="                         SE ESPERA CAMPO NUMERICO
1870    GOTO Disp_error
1880    ENDIF
1890    NEXT I
1900    Input$[1,5]=RPT$(" ",5-LEN(TRIM$(Input$[1,5])))&TRIM$(Input$[1,5])
1910    !
1920    !REVISA LA INFORMACION DEL STATUS Y PONE UN "1" EN CUALQUIER LU-
1930    !GAR QUE TENGA BLANCO O UNO, Y UN CERO CON TECLEAR CUALQUIER COSA
1940    !
1950    FOR I=7 TO 11 STEP 2
1960    IF POS(" 1",Input$[I;1])<>0 THEN
1970    Input$[I;1]="1"
1980    ELSE
1990    Input$[I;1]="0"
2000    ENDIF
2010    NEXT I
2020    !
2030    !ASIGNA LA INFORMACION AL BUFFER
2040    !
2050    Defpunto$[1,5]=Input$[1,5]
2060    !
2070    IF Or$="+X" THEN Defpunto$[18;3]=Input$[11;1]&Input$[9;1]&Input$[7;1]
2080    !
2090    IF Or$="-X" THEN Defpunto$[18;3]=Input$[7;1]&Input$[9;1]&Input$[11;1]
2100    !
2110    IF Or$="+Y" THEN Defpunto$[18;3]=Input$[11;1]&Input$[7;1]&Input$[9;1]
2120    !
2130    IF Or$="-Y" THEN Defpunto$[18;3]=Input$[7;1]&Input$[11;1]&Input$[9;1]
2140    !
2150    !DUPLICA LA INFORMACION EN DEL STATUS_1 EN EL STATUS_2 Y TERMINA
2160    !
2170    Defpunto$[21,23]=Defpunto$[18,20]
2180    IF POS("sS",Input$[13;1])<>0 THEN Defpunto$[27,28]="F1"
2190    GOTO Fin
2200    !
2210    !$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
2220    !$$$$$$                  RUTINAS DE UTILERIA             $$$$$$
2230    !$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
2240    !
2250 Disp_error:
```

```
2260    !
2270    Row1=Row
2280    Col1=Col
2290    GOSUB Limpia_21
2300    PRINT 01;Error$
2310    E-para**
2320    Row=Row1
2330    Col=Col1
2340    GOTO Gera_term
2350    !
2360 Inicia_Limpia:
2370    !
2380    !CALL TADSPCNTL(Term_id,6,,,Status())
2390    !
2400 Limpia:
2410    !
2420    !CALL TADSPCNTL(Term_id,8,,,Status())
2430    RETURN
2440    !
2450 Limpia_21:
2460    !
2470    Row=21
2480    Col=3
2490    GOSUB Fpos
2500    GOSUB Limpia
2510    RETURN
2520    !
2530 Fon
2540    !
2550    !CALL TADSPCNTL(Term_id,26,,,Status())
2560    RETURN
2570    !
2580 Fpos:
2590    !
2600    !CALL TADSPCNTL(Term_id,21,Col,Row,Status())
2610    RETURN
2620    !
2630 Foff:
2640    !
2650    !CALL TADSPCNTL(Term_id,27,,,Status())
2660    RETURN
2670    !
2680 Screen_data_1:DATA "                                                                    "
2690    !
2700            DATA "                          DISTANCIA  RECORRIDA                        "
2710            DATA "                                                                      "
2720            DATA "                              [    ]                                  "
2730            DATA "                                                                      "
2740            DATA "                                                                      "
2750            DATA "                             [ ]                                     "
2760            DATA "                          ▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒                    "
2770            DATA "                          ▒                 ▒▒▒                    "
2780            DATA "                          ▒                   ▒▒                   "
2790            DATA "                          ▒                    ▒▒                  "
```


DATA
Screen_data_2:DATA
FIM (S/N) [ ]
DISTANCIA RECORRIDA
Screen_data_3:DATA
FIM (S/N) [ ]
DISTANCIA RECORRIDA

```
3340              DATA "                    FIN (S/N)  [ ]
3350              DATA "
3360     !
3370 Screen_data_4:DATA "
3380     !
3390              DATA "                      DISTANCIA  RECORRIDA
3400              DATA "
3410              DATA "                           [     ]
3420              DATA "
3430              DATA "                      ################
3440              DATA "                      #              #
3450              DATA "                      #              #
3460              DATA "                    [ ]#              #[ ]
3470              DATA "                      #              #
3480              DATA "                      #              #
3490              DATA "                      #              #
3500              DATA "                      ##            ##
3510              DATA "                       ##          ##
3520              DATA "                        ###      ###
3530              DATA "                          #######
3540              DATA "                            [ ]
3550              DATA "
3560              DATA "
3570              DATA "                    FIN (S/N)  [ ]
3580              DATA "
3590     !
3600 Fin.
3610     !
3620     SUBEND
```

```
1000     !CONTROL SEGMENT POSICION
1010     !CONTROL MAP
1020     !
1030     !
1040     !
1050
1060     !
1070  SUB Pos_orient(Oldvect$,Newvect$)
1080     !
1090     !
1100     !
1110     !
1120     !
1130     !##################################################################
1140     !#####        NUEVA POSICION Y ORIENTACION DEL SIMULADOR       #####
1150     !##################################################################
1160     !
1170     REAL Pos_x
```

```
1100   REAL Pos_y
1110   !
1200   Pos_x=VAL(TRIM$(Oldvec$(6,10)))
1210   IF Oldvec$(11;1)="-" THEN Pos_x=-Pos_x
1220   Pos_y=VAL(TRIM$(Oldvec$(12,16)))
1230   IF Oldvec$(17;1)="-" THEN Pos_y=-Pos_y
1240   !
1250   Newvec$(6,17)=Oldvec$(6,17)
1260   IF Oldvec$(20;1)="1" THEN
1270   IF Oldvec$(27;1)="+" THEN
1280   Pos_x=Pos_x+VAL(TRIM$(Newvec$(1;5)))
1290   ELSE
1300   Pos_x=Pos_x-VAL(TRIM$(Newvec$(1;5)))
1310   ENDIF
1320   Newvec$(11;1)="+"
1330   IF Pos_x<0 THEN Newvec$(11;1)="-"
1340   Newvec$(6,10)=RPT$(" ",5-LEN(TRIM$(VAL$(ABS(Pos_x)))))&TRIM$(VAL$(ABS(Pos_x)))
1350   ELSE
1360   IF Oldvec$(27;1)="+" THEN
1370   Pos_y=Pos_y+VAL(TRIM$(Newvec$(1;5)))
1380   ELSE
1390   Pos_y=Pos_y-VAL(TRIM$(Newvec$(1;5)))
1400   ENDIF
1410   Newvec$(17;1)="+"
1420   IF Pos_y<0 THEN Newvec$(17;1)="-"
1430   Newvec$(12,16)=RPT$(" ",5-LEN(TRIM$(VAL$(ABS(Pos_y)))))&TRIM$(VAL$(ABS(Pos_y)))
1440   ENDIF
1450   Newvec$(24;2)=Oldvec$(27;2)
1460   !
1470   SUBEXD
1480   !
1490   !
1500   !
1510   !
1520   !
1530   !CONTROL SEGMENT DIRECCION
1540   !CONTROL MAP
1550   !
1560   !
1570   !
1580   !
1590   !
1600   SUB Direccion(Vec_dir$)
1610   !
1620   !
1630   !
1640   !
1650   !
1660   !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1670   !!!!!!   DIRECCION DEL SIMULADOR, EN BASE A LA DECISION    !!!!!
1680   !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1690   !
1700   DIM Tabla$(100)
1710   IF Vec_dir$(26,26)="F" THEN
```

```
1720    Vec_dir$[27;2]=Vec_dir$[24;2]
1730    GOTO Fin
1740    END IF
1750    Table$[1,100]="+XD-Y,+XI+Y,-XD+Y,-XI-Y,+YD+X,+YI-X,-YD-X,-YI+X"
1760    Vec_dir$[27,28]=Table$[POS(Table$[1,100],Vec_dir$[24,26])+3;2]
1770    !
1780 Fin:
1790    !
1800    SUBEND
1810    !
1820    !
1830    !
1840    !
1850    !CONTROL SEGMENT CALLEJON
1860    !CONTROL MAP
1870    !
1880    !
1890    !
1900    !
1910    !
1920 SUB Callejon(@1,Vector$(*),INTEGER Contador,Cont_men,Term_id)
1930    !
1940    !
1950    !
1960    !
1970    !
1980    !****************************************************************
1990    !*****          MANEJO DE CALLEJONES SIN SALIDA            *****
2000    !****************************************************************
2010
2020    INTEGER Status(1:2)
2030    !
2040 Checa_callejon:
2050
2060    IF Vector$(Contador)[21,23]<>"000" THEN GOTO Fin
2070    IF Contador=1 THEN GOTO Fin
2080    CALL TMDSPCNTL(Term_id,21,3,21,Status(*))
2090    CALL TMDSPCNTL(Term_id,8,,,Status(*))
2100    PRINT @1;" REGRESO DE "&TRIM$(Vector$(Contador)[1,5])&" METROS Y REORIENTACION"
2110    FOR I=1 TO 20000
2120    NEXT I
2130    Vector$(Contador)[1,28]="                    "
2140    Contador=Contador-1
2150    Cont_men=Cont_men-1
2160    IF Vector$(Contador)[26;1]="D" THEN Vector$(Contador)[21;1]="0"
2170    IF Vector$(Contador)[26;1]="F" THEN Vector$(Contador)[22;1]="0"
2180    IF Vector$(Contador)[26;1]="I" THEN Vector$(Contador)[23;1]="0"
2190    !
2200    Vector$(Contador)[26,28]="   "
2210    Tiep=1
2220    CALL Interfase(@1,Vector$(Contador),Tiep,Term_id)
2230    FOR I=1 TO 20000
2240    NEXT I
2250    GOTO Checa_callejon
```

```
2260    !
2270 Fin:
2280    !
2290    IF Cont_men<0 THEN Cont_men=0
2300    SUBEND
2310    !
2320    !
2330    !
2340    !
2350    !
2360    !
2370    !
2380    !
2390    !
2400    !
2410    $CONTROL SEGMENT DECISION
2420    $CONTROL MAP
2430    !
2440    !
2450    !
2460    !
2470    !
2480 SUB Decision_azar(Vector$)
2490    !
2500    !
2510    !
2520    !
2530    !
2540    !*****************************************************************
2550    !*****         RUTINA DE DECISION AL AZAR                    *****
2560    !*****************************************************************
2570    !
2580    IF Vector$[22;1]="1" THEN
2590    Vector$[26;1]="F"
2600    GOTO Fin
2610    END IF
2620    IF Vector$[21;1]="1" THEN
2630    Vector$[26;1]="I"
2640    GOTO Fin
2650    END IF
2660    IF Vector$[23;1]="1" THEN
2670    Vector$[26;1]="D"
2680    GOTO Fin
2690    END IF
2700    IF 10*(RND)>5 THEN
2710    Vector$[26;1]="D"
2720    ELSE
2730    Vector$[26;1]="I"
2740    END IF
2750    !
2760 Fin:
2770    !
2780    SUBEND
```

```
1000    %CONTROL SEGMENT LOOP
1010    %CONTROL MAP
1020    !
1030    !
1040    !
1050    !
1060    !
1070 SUB Loop(%1,Vector%(),INTEGER Contador,Ns_loop,Cont_men,Term_id)
1080    !
1090    !
1100    !
1110    !
1120    !
1130    !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1140    !%%%%%            RUTINA DE MANEJO DE LOOPS                        %%%%%
1150    !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1160    !
1170    INTEGER Loop_cerrado
1180    INTEGER Igual
1190    INTEGER Close
1200    INTEGER Derecha
1210    INTEGER Status(1:2)
1220    DIM Defpunto%(20)
1230    !
1240    Loop_cerrado=0
1250    !
1260    FOR Igual=1 TO Contador-1
1270    IF Vector%(Contador)(6,17)=Vector%(Igual)(6,17) THEN Si_loop
1280    NEXT Igual
1290    GOTO Fin
1300    !
1310 Si_loop:
1320    !
1330    Fn_loop=1
1340    Derecha=0
1350    FOR Close=Igual+1 TO Contador-1
1360    IF Vector%(Close)(26,1)="F" THEN Next_close_1
1370    IF Vector%(Close)(26,1)="D" THEN
1380    Derecha=Derecha+1
1390    ELSE
1400    Derecha=Derecha-1
1410    ENDIF
1420    !
1430 Next_close_1:
1440    !
1450    NEXT Close
1460    !
1470    FOR Close=Igual+1 TO Contador-1
```

```
1480    IF POS("001,010,100",Vector$(Clase)[21;3])<>0 THEN Next_clase_2
1490    IF Derecha=0 THEN
1500    IF Vector$(Clase)[26;1]="1" THEN Vector$(Clase)[22;1]="0"
1510    IF Vector$(Clase)[26;1]="0" THEN Vector$(Clase)[21;1]="0"
1520    ELSE
1530    IF Vector$(Clase)[26;1]="0" THEN Vector$(Clase)[22;1]="0"
1540    IF Vector$(Clase)[26;1]="1" THEN Vector$(Clase)[23;1]="0"
1550    END IF
1560    IF POS("001,010,100",Vector$(Clase)[21;3])=0 THEN Loop_cerrado=1
1570    !
1580 Next_clase_2:
1590    !
1600    NEXT Clase
1610    !
1620    IF Vector$(Contador)[25;1]=Vector$(Igual)[25;1] THEN
1630    Vector$(Igual)[22;1]="0"
1640    IF Loop_cerrado=1 THEN
1650    Vector$(Contador)[21;3]="100"
1660    IF Vector$(Igual)[24;1]=Vector$(Contador)[24;1] THEN
1670    IF Derecha=0 THEN
1680    Vector$(Igual)[21;1]="0"
1690    ELSE
1700    Vector$(Igual)[23;1]="0"
1710    END IF
1720    END IF
1730    GOTO Fin
1740    END IF
1750    IF Vector$(Igual)[24;1]<>Vector$(Contador)[24;1] THEN
1760    GOSUB Limpia_21
1770    PRINT @1;"   DOS VUELTAS A LA DERECHA "
1780    FOR I=1 TO 20000
1790    NEXT I
1800    CALL Interfase(@1,Vector$(Igual),1,Term_id)
1810    IF Derecha=0 THEN
1820    Vector$(Igual)[23;1]="0"
1830    ELSE
1840    Vector$(Igual)[21;1]="0"
1850    END IF
1860    ELSE
1870    IF Derecha=0 THEN
1880    Vector$(Igual)[21;1]="0"
1890    ELSE
1900    Vector$(Igual)[23;1]="0"
1910    END IF
1920    END IF
1930    Vector$(Igual)[26,28]="   "
1940    Cont_mem=Cont_mem-(Contador-Igual)
1950    Contador=Igual
1960    GOTO Fin
1970    ELSE
1980    IF Vector$(Contador)[24;2]=Vector$(Igual)[27;2] THEN
1990    IF Loop_cerrado=0 THEN
2000    IF (Derecha=0 AND Vector$(Igual)[26;1]="1") OR (Derecha<>0 AND Vector$(Igual)[26;1]="0") THEN
2010    Vector$(Igual)[21;3]="000"
```

```
2020    ELSE
2030    Vector$[Iposl)[21;1]="0"
2040    Vector$[Iposl)[23;1]="1"
2050    END IF
2060    !
2070    GOSUB Limpio_21
2080    IF Vector$[Iposl)[26;1]="1" THEN
2090    PRINT @1;" VUELTA A LA DERECHA "
2100    ELSE
2110    PRINT @1;" VUELTA A LA IZQUIERDA "
2120    END IF
2130    !
2140 Delay:
2150    !
2160    FOR I=1 TO 20000
2170    NEXT I
2180    CALL Interfase(@1,Vector$[Iposl),1,Term_id)
2190    Cont_new=Cont_new-(Contador-Iposl)
2200    Contador=Iposl
2210    GOTO Fin
2220    ELSE
2230    !
2240 If_derecho:
2250    !
2260    IF Derecho)1 THEN
2270    Vector$[Iposl)[21;1]="1"
2280    ELSE
2290    Vector$[Iposl)[23;1]="1"
2300    END IF
2310    Vector$[Contador)[21;3]="000"
2320    GOTO Fin
2330    END IF
2340    ELSE
2350    IF Lazo_cerrado()0 THEN If_derecho
2360    Vector$[Iposl)[22;1]="1"
2370    GOSUB Limpio_21
2380    IF Derecho)1 THEN
2390    Vector$[Iposl)[21;1]="1"
2400    PRINT @1;" VUELTA A LA DERECHA "
2410    ELSE
2420    Vector$[Iposl)[23;1]="1"
2430    PRINT @1;" VUELTA A LA IZQUIERDA "
2440    END IF
2450    GOTO Delay
2460    END IF
2470    END IF
2480    !
2490 Limpio_21:
2500    !
2510    !CALL TRDSPCNTL(Term_id,21,3,21,Status())
2520    !CALL TRDSPCNTL(Term_id,6,,,,Status())
2530    RETURN
2540    !
2550 Fin:
```

```
2560    !
2570    Vector$(Contador+1)[1,20]=""
2580    IF Cont_mem<0 THEN Cont_mem=1
2590    SUBEND
```

```
1000    %CONTROL SEGMENT OLVIDO
1010    %CONTROL MAP
1020    !
1030    !
1040    !
1050    !
1060    !
1070 SUB Olvido(01,INTEGER Term_id)
1080    !
1090    !
1100    !
1110    !
1120    !
1130    INTEGER  Status(1:2)
1140    INTEGER  K
1150    DIM Line$[220]
1160    DIM Input$[1]
1170    DIM Patpan$[3]
1180    !
1190 Screen:
1200    !
1210    ICALL TRDSPCNTL(Term_id,6,,,Status()) !Posiciono el cursor en la esquina superior izquierda
1220    ICALL TRDSPCNTL(Term_id,8,,,Status()) !Limpia todo lo que este abajo del cursor
1230    RESTORE Screen_data
1240    FOR   K=0 TO 20
1250    Line$[1,220] = ""
1260    READ Line$
1270    PRINT @1; Line$
1280    NEXT K
1290    !
1300 Opcion:
1310    !
1320    ICALL TRDSPCNTL(Term_id,24,3,21,Status()) !Posiciono el cursor. Row=24 Col=3
1330    ICALL TRDSPCNTL(Term_id,8,,,Status()) !Limpia todo lo que este abajo del cursor
1340    PRINT @1;     "              Teclee la opciFn deseada y luego 'ENTER'"
1350    Patpan$[1,3]="  N"
1360    ICALL TRDSPCNTL(Term_id,26,,,Status()) !Acceso a la terminal como archivo
1370    ICALL PUTN(Term_id,Patpan$,LEN(Patpan$),Status())
1380    ICALL TRDSPCNTL(Term_id,27,,,Status()) !Acceso a terminal default
1390    Input$[1,1]=""
1400    ICALL TRDSPCNTL(Term_id,26,,,Status()) !Acceso a la terminal como archivo
1410    ICALL TRDSPCNTL(Term_id,21,37,14,Status()) !Posiciono el cursor en la pantalla
1420    ICALL GETN(Term_id,Input$,LEN(Input$),Status())
```

```
1430   !CALL TADSPCNTL(Term_id,27,,,Status()) !Acceso a terminal defoult
1440   IF FOS("Ss",Input$(1),1)=0 THEN Fin
1450   !
1460   ! EN CASO DE QUE LA RESPUESTA SEA SI, RECREEA LOS ARCHIVOS
1470   !
1480   ! ARCHIVO GUIA
1490   !
1500   Arguia$="TESGUIA(1)TESIS)"
1510   !CALL PURGE(Arguia$,LEN(Arguia$),Status())
1520   !CALL CREATE(Arguia$,LEN(Arguia$),Status(),"K",,,20,50,"C",4)
1530   !
1540   ! ARCHIVO DE RUTAS
1550   !
1560   Arruta$="TESRUTA(1)TESIS)"
1570   !CALL PURGE(Arruta$,LEN(Arruta$),Status())
1580   !CALL CREATE(Arruta$,LEN(Arruta$),Status(),"K",,,20000,78,"C",4,,1)
1590   !
1600   GOTO Fin
1610   !              01234567891-1234567891-1234567891-1234567891-1234567891-1234567891-1234567891-12345
1620   !
1630 Screin_data:DATA *                                                                                  *
1640   !
1650           DATA *                                                                                   *
1660           DATA *                                                                                   *
1670           DATA *                                                                                   *
1680           DATA *                                                                                   *
1690           DATA *                  RUTINA DE OLVIDO Y RECUERDO DE RUTAS ANTERIORES                  *
1700           DATA *                                                                                   *
1710           DATA *                                                                                   *
1720           DATA *                                                                                   *
1730           DATA *                                                                                   *
1740           DATA *                                                                                   *
1750           DATA *                                                                                   *
1760           DATA *                   DESEA BORRAR LAS RUTAS ANTERIORES ? (S/N)                       *
1770           DATA *                                                                                   *
1780           DATA *                                  [ ]                                              *
1790           DATA *                                                                                   *
1800           DATA *                                                                                   *
1810           DATA *                                                                                   *
1820           DATA *                                                                                   *
1830           DATA *                                                                                   *
1840           DATA *                                                                                   *
1850   !
1860 Fin:
1870   !
1880   SUBEND


1890   !CONTROL SEGMENT MEMORIA
```

```
1010    %CONTROL MAP
1020    !
1030    !
1040    !
1050    !
1060    !
1070 SUB Memoriza(A1,Vector$(),INTEGER Contador,Num_ruta,Vec_men,Pos_men,Pos_vec,Term_id)
1080    !
1090    !
1100    !
1110    !
1120    IF Pos_vec=1 THEN !in
1130    !
1140    INTEGER Argsio_id
1150    INTEGER Arruta_id
1160    INTEGER I
1170    INTEGER Status(1:2)
1180    !
1190    !CALL TRDSPCNTL(Term_id,21,3,21,Status()) !Posiciona el cursor Col=3 Row=21
1200    !CALL TRDSPCNTL(Term_id,0,,,Status()) !Limpia el renglon y abajo
1210    PRINT @1;     "                              MEMORIZACION EN PROCESO
1220    !
1230    DIM Buf_gsio$[58]
1240    DIM Buf_ruta$[70]
1250    DIM Num_ruta$[4]
1260    DIM Num_reg$[10]
1270    DIM Num_gtos$[10]
1280    DIM Vec_men$(4)
1290    DIM Pos_men$(10)
1300    !
1310    IF Pos_men(0) THEN Pos_men=Pos_men+1
1320    !
1330    Argsio$="TESCUTA()JTESIS)"
1340    Arruta$="TESRUTA()JTESIS)"
1350    !CALL OPEN(Argsio$,LEN(Argsio$),Argsio_id,Status())
1360    !CALL OPEN(Arruta$,LEN(Arruta$),Arruta_id,Status())
1370    !
1380    Num_ruta=Num_ruta+1
1390    IF Num_ruta>11 THEN
1400    Num_ruta=10
1410    Num_ruta$(1,4)=RPT$(" ",4-LEN(TRIM$(VAL$(Num_ruta))))&TRIM$(VAL$(Num_ruta))
1420    !CALL DELETE(Argsio_id,Num_ruta$,Status())
1430    !
1440 Delete_ruta:
1450    !
1460    !CALL DELETE(Arruta_id,Num_ruta$,Status())
1470    IF Status(1)=0 THEN Delete_ruta
1480    ENDIF
1490    !
1500    ! ASIGNA EL BUFFER GUIA Y LO PONE EN EL ARCHIVO
1510    !
1520    Num_ruta$(1,4)=RPT$(" ",4-LEN(TRIM$(VAL$(Num_ruta))))&TRIM$(VAL$(Num_ruta))
1530    IF Pos_vec=0 THEN
1540    Num_reg$(1,10)=RPT$(" ",10-LEN(TRIM$(VAL$(Contador))))&TRIM$(VAL$(Contador))
```

```
1550    ELSE
1560    Nom_reg$[1,10]=RPT$(' ',10-LEN(TRIM$(VAL$(Pas_vec))))&TRIM$(VAL$(Pas_vec))
1570    END IF
1580    Nom_ptos$[1,10]=RPT$(' ',10-LEN(TRIM$(VAL$(Contador))))&TRIM$(VAL$(Contador))
1590    !
1600    Def_guia$[1,50]=""
1610    Def_guia$[1,50]=Nom_ruta$[1,4]&Nom_reg$[1,10]&Nom_ptos$[1,10]
1620    !CALL PUT(Arguia_id,Def_guia$,LEN(Def_guia$),Nom_ruta$[1,4],Status())
1630    !
1640    IF Pas_vec()0 THEN Contador=Pas_vec
1650    !
1660    FOR I=1 TO Contador
1670    !
1680    ! ASIGNA EL BUFFER DE LA RUTA Y LO POME EN EL ARCHIVO
1690    !
1700    Def_ruta$[1,76]=""
1710    Def_ruta$[1,76]=Nom_ruta$[1,4]&Vector$(I)[1,28]
1720    IF I=Pas_vec THEN
1730    Vec_men$[1,4]=RPT$(' ',4-LEN(TRIM$(VAL$(Vec_men))))&TRIM$(VAL$(Vec_men))
1740    Pas_men$[1,10]=RPT$(' ',10-LEN(TRIM$(VAL$(Pas_men))))&TRIM$(VAL$(Pas_men))
1750    Def_ruta$[33,70]=Vec_men$[1,4]&Pas_men$[1,10]
1760    END IF
1770    IF I=Contador THEN Def_ruta$[47;1]="F"
1780    !CALL PUT(Arruta_id,Def_ruta$,LEN(Def_ruta$),Nom_ruta$[1,4],Status())
1790    NEXT I
1800    !
1810    !CALL CLOSE(Arguia_id,Status())
1820    !CALL CLOSE(Arruta_id,Status())
1830    !
1840 Fin:
1850    !
1860    SUBEXB
```

```
1000    %CONTROL SEGMENT RECORDAR
1010    %CONTROL MAP
1020    !
1030    !
1040    !
1050    !
1060    !
1070 SUB Recordar(01,V1$(),V2$(),V3$(),V4$(),V5$(),V6$(),V7$()
                  V8$(),V9$(),V0$(),Vec_guia$(),INTEGER Cola,Term_id)
1080    !
1090    !
1100    !
1110    !
1120    !
1130    INTEGER Status(1;2)
```

```
1140    INTEGER I,J,K,L
1150    INTEGER Arquio_id
1160    INTEGER Arrota_id
1170    INTEGER Ruta_conex
1180    INTEGER Conexion
1190    INTEGER Fin_conex
1200    !
1210    ICALL TADSPCNTL(Term_id,21,3,21,Status()) !Posiciona el cursor Col=3 Row=21
1220    ICALL TADSPCNTL(Term_id,8,,,Status()) !Limpia el renglon y abajo
1230    PRINT #1;     *                SIMULADOR RECORRIENDO LAS RUTAS ANTERIORES
1240    !
1250    DIM Befquio$[50]
1260    DIM Befrota$[70]
1270    DIM Vec_quio$[1:2000][20]
1280    DIM Vec_qui_men$[1:2000][20]
1290    !
1300    !
1310    Arquio$="ESCUTA(1)TESIS"
1320    Arrota$="ESRUTA(1)TESIS"
1330    ICALL OPEN(Arquio$,LEN(Arquio$),Arquio_id,Status())
1340    ICALL OPEN(Arrota$,LEN(Arrota$),Arrota_id,Status())
1350    !
1360    FOR Guio=1 TO 10
1370    Befquio$[1,50]=""
1380    ICALL GETN(Arquio_id,Befquio$,LEN(Befquio$),Status())
1390    IF Status(1)<0 THEN
1400    IF Guio<>1 THEN Asigno_rutas
1410    Guio=0
1420    GOTO Fin
1430    ENDIF
1440    Vec_quio$(Guio)[1,50]=Befquio$[1,50]
1450    NEXT Guio
1460    !
1470 Asigno_rutas:
1480    !
1490    Guio=Guio-1
1500    !
1510    FOR I=1 TO Guio
1520    FOR J=1 TO VAL(Vec_quio$(I)[5,14])
1530    Befrota$[1,70]=""
1540    ICALL GETN(Arrota_id,Befrota$,LEN(Befrota$),Status())
1550    Vec_qui$(J)[1,20]=Befrota$[5,32]
1560    NEXT J
1570    IF I=1 THEN Lleno_vectores
1580    IF Vec_quio$(I)[5,14]=Vec_quio$(I)[15,24] THEN Lleno_vectores
1590    Ruta_conex=VAL(Befrota$[33,36])
1600    Conexion=VAL(Befrota$[37,46])
1610    Fin_conex=VAL(Vec_quio$(Ruta_conex)[15,24])
1620    IF Ruta_conex=1 THEN MAT Vec_qui_men$=V1$
1630    IF Ruta_conex=2 THEN MAT Vec_qui_men$=V2$
1640    IF Ruta_conex=3 THEN MAT Vec_qui_men$=V3$
1650    IF Ruta_conex=4 THEN MAT Vec_qui_men$=V4$
1660    IF Ruta_conex=5 THEN MAT Vec_qui_men$=V5$
1670    IF Ruta_conex=6 THEN MAT Vec_qui_men$=V6$
```

```
1680    IF Ruta_conex=7 THEN MAT Vec_aux_mem1=V78
1690    IF Ruta_conex=8 THEN MAT Vec_aux_mem1=V88
1700    IF Ruta_conex=9 THEN MAT Vec_aux_mem1=V98
1710    IF Ruta_conex=10 THEN MAT Vec_aux_mem1=V108
1720    FOR K=Conexion TO Fin_conex
1730    Vec_aux$(J)=""
1740    Vec_aux$(J)[1,28]=Vec_aux_mem1$(K)[1,5]
1750    CALL Pos_orient(Vec_aux$(J-1),Vec_aux$(J))
1760    Vec_aux$(J)[10,23]=Vec_aux_mem1$(K)[10,23]
1770    Vec_aux$(J)[26,26]=Vec_aux_mem1$(K)[26,26]
1780    IF Vec_aux_mem1$(K)[27,28]="FI" THEN
1790    Vec_aux$(J)[27,28]="FI"
1800    ELSE
1810    CALL Direccion(Vec_aux$(J))
1820    END IF
1830    J=J+1
1840    NEXT K
1850    !
1860 Llena_vectores:
1870    !
1880    FOR L=1 TO VAL(Vec_aux1$(1)[15,24])
1890    IF I=1 THEN V18(L)=Vec_aux$(L)
1900    IF I=2 THEN V28(L)=Vec_aux$(L)
1910    IF I=3 THEN V38(L)=Vec_aux$(L)
1920    IF I=4 THEN V48(L)=Vec_aux$(L)
1930    IF I=5 THEN V58(L)=Vec_aux$(L)
1940    IF I=6 THEN V68(L)=Vec_aux$(L)
1950    IF I=7 THEN V78(L)=Vec_aux$(L)
1960    IF I=8 THEN V88(L)=Vec_aux$(L)
1970    IF I=9 THEN V98(L)=Vec_aux$(L)
1980    IF I=10 THEN V108(L)=Vec_aux$(L)
1990    Vec_aux$(L)=""
2000    NEXT L
2010    NEXT I
2020    !
2030 Fin
2040    !
2050    !CALL CLOSE(Argoin_id,Status(1))
2060    !CALL CLOSE(Arrota_id,Status(1))
2070    !END


1000    %CONTROL SEGMENT BUSCA
1010    %CONTROL MAP
1020
1030    !
1040
1050    !
1060    !
```

```
1070 SUB Busca(V1$(),V2$(),V3$(),V4$(),V5$(),V6$(),V7$(),V8$(),V9$(),V0$(),
             Vec_guia$(),Vector$(),INTEGER Guia,Cont,Vec_men,Pos_men,Pos_vec,Cont_men,Term_id)
1080   !
1090   !
1100   !
1110   !
1120   !
1130   INTEGER Fin_cones
1140   !
1150   DIM Vec_aux$(1:200)[20]
1160   !
1170   Cont_men=Cont_men+1
1180   !
1190 Busca_guia:
1200   !
1210   FOR Vec_men=1 TO Guia
1220   Fin_cones=VAL(Vec_guia$(Vec_men)[15,24])
1230   IF Fin_cones(Cont_men THEN Next_vec_men
1240   !
1250   IF Vec_men=1 THEN MAT Vec_aux$=V1$
1260   IF Vec_men=2 THEN MAT Vec_aux$=V2$
1270   IF Vec_men=3 THEN MAT Vec_aux$=V3$
1280   IF Vec_men=4 THEN MAT Vec_aux$=V4$
1290   IF Vec_men=5 THEN MAT Vec_aux$=V5$
1300   IF Vec_men=6 THEN MAT Vec_aux$=V6$
1310   IF Vec_men=7 THEN MAT Vec_aux$=V7$
1320   IF Vec_men=8 THEN MAT Vec_aux$=V8$
1330   IF Vec_men=9 THEN MAT Vec_aux$=V9$
1340   IF Vec_men=10 THEN MAT Vec_aux$=V0$
1350   FOR I=Cont_men TO Fin_cones
1360   FOR K=1 TO Cont_men
1370   IF (Vector$(Cont-Cont_men+I)[1,5]()Vec_aux$(J-Cont_men+K)[1,5]) OR
          (Vector$(Cont-Cont_men+I)[10,20]()Vec_aux$(J-Cont_men+K)[10,20]) THEN Next_j
1380   IF Vector$(Cont-Cont_men+K)[10,20]()Vector$(Cont-Cont_men+I)[21,23] THEN Next_j
1390   NEXT K
1400   GOTO Asigna_men
1410   !
1420 Next_j:
1430   !
1440   NEXT J
1450   !
1460 Next_vec_men:
1470   !
1480   NEXT Vec_men
1490   Cont_men=Cont_men-1
1500   IF Cont_men=0 THEN Fin
1510   GOTO Busca_guia
1520   !
1530 Asigna_men:
1540   !
1550   Pos_vec=Cont-Cont_men+1
1560   Pos_men=J-Cont_men+1
1570   Vector$(Cont)[26,26]=Vec_aux$(Pos_men+Cont_men-1)[26,26]
1580   CALL Direccion(Vector$(Cont))
```

```
1590    IF Cont_men(1) THEN Fin
1600    Vector$(Cont+1)[1,5]=Vec_aux$(Pos_men+Cont_men)[1,5]
1610    Vector$(Cont+1)[18,20]=Vec_aux$(Pos_men+Cont_men)[18,20]
1620    Vector$(Cont+1)[21,23]=Vec_aux$(Pos_men+Cont_men)[18,20]
1630    !
1640 Fin:
1650    !
1660    IF Cont_men=0 THEN Vec_men,Pos_men,Pos_vec=0
1670    SUBEND
1680    !
1690    !
1700    !
1710    !
1720    !
1730    !
1740    !
1750    !
1760    !
1770    !
1780    !
1790    !
1800    !CONTROL SEGMENT SERVICIO
1810    !CONTROL MAP
1820    !
1830    !
1840    !
1850    !
1860    !
1870 SUB Servicio(@i,Otro$,V10(),V20(),V30(),V40(),V50(),V60(),V70(),V80(),
                  V90(),V00(),Vector$(),Vec_grua$(),INTEGER Contador,Cosa,Term_id)
1880    !
1890    !
1900    !
1910    !
1920    !
1930    !
1940    INTEGER Status(1:2)
1950    INTEGER K
1960    INTEGER Row
1970    INTEGER Col
1980    INTEGER Men_ptos
1990    INTEGER Men_rota
2000    !
2010    DIM Pregun$(6)
2020    DIM Input$(8)
2030    DIM Line$(220)
2040    DIM Vec_aux$(1:2000)[20]
2050    !
2060 Screen:
2070    !
2080    GOSUB Inicio_limpio
2090    RESTORE Screen_data
2100    FOR   K=0 TO 20
2110    Line$(1,220) = ""
2120    READ Line$
2130    PRINT @1; Line$
2140    NEXT K
2150    !
2160 Opcion:
```

```
1370   !
1380   GOSUB Limpio_21
1390   PRINT @1;"                Teclee la opcifn deseada y luego 'ENTER'"
1400   Patpun$[1,6]="  F =N"
1410   GOSUB Fon
1420   !CALL PUTN(Term_id,Patpun$,LEN(Patpun$),Status())
1430   GOSUB Foff
1440   Row=10
1450   Col=49
1460   !
1470 Getn_term:
1480   !
1490   Input$[1,8]=""
1500   GOSUB Fon
1510   GOSUB Fpos
1520   !CALL GETN(Term_id,Input$,LEN(Input$),Status())
1530   GOSUB Foff
1540   !
1550   IF POS("FF",TRIM$(Input$[1,2]))<>0 THEN
1560   GOSUB Limpio_21
1570   PRINT @1;"              DESEA RESOLVER OTRO LABERINTO ? (S/N)  (S)"
1580   Row=21
1590   Col=56
1600   Input$[1,8]=""
1610   GOSUB Fon
1620   GOSUB Fpos
1630   !CALL GETN(Term_id,Input$,LEN(Input$),Status())
1640   GOSUB Foff
1650   Otro$[1,1]="N"
1660   IF POS("S",Input$[0,0])<>0 THEN Otro$[1,1]="S"
1670   GOTO Fin
1680   END!F
1690   IF POS("1,2,3,4,5,6,7,8,9,10",TRIM$(Input$[1,2]))=0 THEN
1700   GOSUB Limpio_21
1710   PRINT @1;"                        Teclee solamente 1-10"
1720   Row=10
1730   Col=49
1740   GOTO Getn_term
1750   END!F
1760   IF VAL(Input$[1,2])<>Coin THEN
1770   MAT Vec_aux$=Vector$
1780   ELSE
1790   IF VAL(Input$[1,2])=1 THEN MAT Vec_aux$=V1$
1800   IF VAL(Input$[1,2])=2 THEN MAT Vec_aux$=V2$
1810   IF VAL(Input$[1,2])=3 THEN MAT Vec_aux$=V3$
1820   IF VAL(Input$[1,2])=4 THEN MAT Vec_aux$=V4$
1830   IF VAL(Input$[1,2])=5 THEN MAT Vec_aux$=V5$
1840   IF VAL(Input$[1,2])=6 THEN MAT Vec_aux$=V6$
1850   IF VAL(Input$[1,2])=7 THEN MAT Vec_aux$=V7$
1860   IF VAL(Input$[1,2])=8 THEN MAT Vec_aux$=V8$
1870   IF VAL(Input$[1,2])=9 THEN MAT Vec_aux$=V9$
1880   IF VAL(Input$[1,2])=10 THEN MAT Vec_aux$=V0$
1890   END!F
1900   IF Input$[1,2]="10" THEN Input$[1,2]="0"
```

```
1910    Num_ptos=Contador
1920    IF VAL(Input$(1,2))<=Cuio THEN Num_ptos=VAL(Vec_guio$(VAL(Input$(1,2)))(15;24))
1930    IF POS("Si",Input$(6,6))<>0 THEN
1940    GOSUB Limpio_21
1950    PRINT @1;      '                        IMPRESION EN PROCESO
1960    Num_ptos=VAL(Input$(1,2))
1970    CALL Imprime(@1,Vec_guio$(),Num_ptos,Num_ruta,Term_id)
1980    ENDIF
1990    IF POS("Fi",Input$(4,4))<>0 THEN CALL Frente(@1,Vec_guio$(),Num_ptos,Term_id)
2000    IF POS("Re",Input$(4,4))<>0 THEN CALL Regreso(@1,Vec_guio$(),Num_ptos,Term_id)
2010    IF POS("FiNi",Input$(4,4))<>0 THEN GOTO Screen
2020    GOTO Opcion
2030    !
2040    !            @123456789-123456789-123456789-123456789-123456789-123456789-123456789-123456789-1234
2050 Screen_data:DATA *                                                                                  *
2060    !
2070            DATA *                                                                                   *
2080            DATA *                                                                                   *
2090            DATA *                                                                                   *
2100            DATA *                                                                                   *
2110            DATA *                              RUTINA DE SERVICIOS                                  *
2120            DATA *                                                                                   *
2130            DATA *                                                                                   *
2140            DATA *                                                                                   *
2150            DATA *                                                                                   *
2160            DATA *                                 RUTA ? (1-10)  [  ]                               *
2170            DATA *                                                                                   *
2180            DATA *                          FRENTE O REGRESO ? (F/R)  [ ]                            *
2190            DATA *                                                                                   *
2200            DATA *                                 IMPRESION ? (S/N)  [ ]                            *
2210            DATA *                                                                                   *
2220            DATA *                                                                                   *
2230            DATA *                                                                                   *
2240            DATA *                                                                                   *
2250            DATA *                                                                                   *
2260            DATA *                                                                                   *
2270    !
2280 Inicio_Limpio:
2290    !
2300    !CALL TADSPCNTL(Term_id,6,,,Status())
2310    !
2320 Limpio:
2330    !
2340    !CALL TADSPCNTL(Term_id,0,,,Status())
2350    RETURN
2360    !
2370 Limpio_21:
2380    !
2390    Row=21
2400    Col=3
2410    GOSUB Fpos
2420    GOSUB Limpio
2430    RETURN
2440    !
```

```
2450 Fen:
2460   !
2470   !CALL TRDSPCNTL(Term_id,26,,,Status())
2480   RETURN
2490   !
2500 Fpos:
2510   !
2520   !CALL TRDSPCNTL(Term_id,21,Col,Row,Status())
2530   RETURN
2540   !
2550 Feff:
2560   !
2570   !CALL TRDSPCNTL(Term_id,27,,,Status())
2580   RETURN
2590   !
2600 Fins:
2610   !
2620   SUSEND
2630   !
2640   !
2650   !
2660   !
2670   !
2680   %CONTROL SEGMENT IMPRIME
2690   %CONTROL MAP
2700   !
2710   !
2720   !
2730   !
2740   !
2750 SUB Imprime(@1,Vec_cue$(),INTEGER Num_ptas,Num_ruta,Term_id)
2760   !
2770   !
2780   !
2790   !
2800   !
2810   !
2820   DIM Out$[134]
2830   INTEGER Status(1:2)
2840   INTEGER Out_id
2850   INTEGER I
2860   !
2870   ! ABRE LA IMPRESORA .
2880   ASSIGN @8 TO *:PRINTER"
2890   !CALL FOFILEID(8),Out_id
2900   !
2910   Out$[1,134]=""
2920   Out$[1,134]="1"
2930   GOSUB Pot
2940   Out$[1,134]="                              IMPRESION DE LA RUTA "&TRIM$(VAL$(Num_ruta))
2950   GOSUB Pot
2960   GOSUB Pot
2970   Out$[1,134]="                                   STATUS 1    STATUS 2"
2980   GOSUB Pot
```

```
2990    !             123456789-123456789-123456789-123456789-123456789-123456789-123456789-123456789-123456789-12345678
3000    Out$(1,134)="      DISTANCIA     POS_X      POS_Y   D  F  I     D  F  I      ORIENTACION      VUELTA      DIRECCION"
3010    !                  #####       #####.     #####.  #  #  #     #  #  #          ##              #            ##
3020    GOSUB Put
3030    GOSUB Put
3040    Out$(3,134)=RPT$("=",100)
3050    GOSUB Put
3060    GOSUB Put
3070    FOR I=1 TO Num_ptos
3080    Out$(8,134)=Vec_out$(I)[1,5]
3090    Out$(20,134)=Vec_out$(I)[6,11]
3100    Out$(30,134)=Vec_out$(I)[12,17]
3110    Out$(38,134)=Vec_out$(I)[18;1]&"  "&Vec_out$(I)[19;1]&"  "&Vec_out$(I)[20;1]
3120    Out$(50,134)=Vec_out$(I)[21;1]&"  "&Vec_out$(I)[22;1]&"  "&Vec_out$(I)[23;1]
3130    Out$(66,134)=Vec_out$(I)[24;2]
3140    Out$(80,134)=Vec_out$(I)[26;1]
3150    Out$(93,134)=Vec_out$(I)[27;2]
3160    GOSUB Put
3170    GOSUB Put
3180    NEXT I
3190    GOTO Fin
3200    !
3210 Put:
3220    .
3230    FOR Z=220 TO 5 STEP -1
3240    IF Out$[Z,Z]<>" " THEN Putm
3250    NEXT Z
3260
3270 Putm:
3280    .
3290    Out$ = Out$[1,Z]
3300    CALL PUTN(Out_id, Out$, LEN(Out$),Status(*))
3310    Out$[1,220] = " "
3320    Num_lines=Num_lines+1
3330    RETURN
3340
3350 Fin.
3360
3370    ASSIGN @0 TO *
3380    SUBEND
3390
3400    !
3410    !
3420    !
3430    !
3440    $CONTROL SEGMENT FRENTE
3450    $CONTROL MAP
3460    !
3470    !
3480    !
3490    !
3500    !
3510 SUB Frente(@1,Vec_out$(*),INTEGER Num_ptos,Term_id)
3520    !
```

```
3530    !
3540    !
3550    !
3560    !
3570    !
3580    DIM Defpuntos(20)
3590    INTEGER I
3600    !
3610    Distancia=0
3620    FOR I=1 TO Num_ptos
3630    IF I=1 THEN
3640    Defpuntos=Vec_ant(1)
3650    GOTO Call_interfase
3660    END IF
3670    Distancia=Distancia+VAL(Vec_ant$(I)[1,5])
3680    IF Vec_ant$(I)[26,26]="F" THEN Next_i
3690    Defpuntos$(1,5)=RPT$(" ",5-LEN(TRIM$(VAL$(Distancia))))&TRIM$(VAL$(Distancia))
3700    Defpuntos$(6,20)=Vec_ant$(I)[6,20]
3710    !
3720 Call_interfase:
3730    !
3740    Distancia=0
3750    CALL Interfase(01 Defpuntos$,1,Term_id)
3760    FOR J=1 TO 50000
3770    NEXT J
3780    !
3790 Next_i:
3800    !
3810    NEXT I
3820    SUBEND
3830    !
3840    !
3850    !
3860    !
3870    !
3880    %CONTROL SEGMENT REGRESO
3890    %CONTROL MAP
3900    !
3911    !
3920    !
3930    !
3940    !
3950 SUB Regreso(01,Vec_ant$(),INTEGER Num_ptos,Term_id)
3960    !
3970    !
3980    !
3990    !
4000    !
4010    !
4020    INTEGER I
4030    !
4040    FOR I=1 TO Num_ptos
4153    Vec_ant$(I)[21,23]="000"
4160    NEXT I
```

```
4076      CALL Interface(01,Vec_ass$(Non_ptss),1,Term_id)
4088      CALL Collejan(01,Vec_ass$(),Non_ptss,0,Term_id)
4091      SUBEND
```

9.- RESULTADOS

EN EL MOMENTO EN QUE EL SIMULADOR ESTUVO EN CONDICIONES DE TRABAJAR, SE EMPEZARON A REALIZAR PRUEBAS, Y SE PUEDE AFIRMAR QUE EL RESULTADO DE LAS MISMAS INDICA QUE SE TUVO EXITO, PUES EL SIMULADOR CUMPLIA PERFECTAMENTE CON LOS OBJETIVOS DESEADOS.

UNO DE LOS HECHOS MAS SORPRENDENTES, Y QUE EN VERDAD NO SE ESPERABA ES EN LO QUE SE REFIERE A EL RECUERDO DE RUTAS ANTERIORES. SE SUPONE QUE UN SER HUMANO MIENTRAS MAS SEGURO ESTE DE QUE HA RECONOCIDO UN CAMINO, PUEDE RECORRER ESTE CON MAYOR FACILIDAD. ASI MISMO, EL SIMULADOR MIENTRAS MAS PUNTOS CONOCIDOS SE VA ENCONTRANDO EN UNA RUTA, EMPIEZA A TOMAR LAS DECISIONES CON MAS RAPIDEZ. ESTO COMO SE DIJO ANTES, PROVOCO CIERTA SORPRESA, PUES NO SE HABIA PREVISTO ESTA REACCION. SIN EMBARGO DESPUES DE ANALIZAR LA RAZON DE ESTE COMPORTAMIENTO SE ENCONTRO UNA EXPLICACION MUY SIMPLE.

EL HECHO ES QUE CUANDO EL SIMULADOR SE HA ENCONTRADO UNA RUTA CONOCIDA, NO SIGUE ESTA RUTA A CIEGAS, SINO QUE POR EL CONTRARIO, SIGUE REVISANDO PUNTO POR PUNTO CADA VEZ QUE AVANZA UN PASO, DE ESTA MANERA SE PROTEGE CONTRA POSIBLES EQUIVOCACIONES. EN UN PRINCIPIO SE PENSO QUE ESTO HARIA QUE EL SIMULADOR FUERA MAS LENTO CADA VEZ, PERO SUCEDE LO CONTRARIO, Y ESTO ES PORQUE CADA VEZ QUE AVANZA UN PASO Y QUIERA CONFIRMAR QUE ESTA EN LA RUTA PENSADA, TIENE QUE REVISAR UN MAYOR NUMERO DE PUNTOS CONCORDANTES, PERO A SU VEZ ESTO LIMITA EL NUMERO DE REVISIONES QUE SE TIENEN QUE HACER EN UNA RUTA ERRONEA, PUES ESTE NUMERO SE VA INCREMENTANDO MIENTRAS EL NUMERO DE PUNTOS A REVISAR ES MENOR A LA MITAD DEL NUMERO TOTAL DE PUNTOS EN LA RUTA, MIENTRAS QUE AL PASAR DE ESTE PUNTO, EL NUMERO DE REVISIONES SE DECREMENTA A LA MISMA VELOCIDAD QUE SE INCREMENTO, HASTA QUE LLEGADO EL MOMENTO EN QUE EL NUMERO DE PUNTOS A COMPARAR ES MAYOR QUE EL NUMERO DE PUNTOS DE LA RUTA; EN ESTA SITUACION NI SIQUIERA SE COMPARA ESTA RUTA, SIMPLEMENTE SE SALTA.

POR EJEMPLO:

SUPONGAMOS QUE TENEMOS DOS RUTAS. LA PRIMERA TIENE UN TOTAL DE 30 PUNTOS, Y LA SEGUNDA UN TOTAL DE 50. EL SIMULADOR HA RECONOCIDO QUE ENTRO EN LA SEGUNDA RUTA, EN EL PASO 15, Y LLEVA 10 PUNTOS QUE CONCUERDAN PERFECTAMENTE. PARA REVISAR ESOS 10 PUNTOS TUVO QUE VER PRIMERO LA RUTA 1, POR LO CUAL REALIZO LA COMPARACION DE ESTOS 10 EN 21 POSICIONES DIFERENTES EN LA RUTA 1. ESTO NOS DA UN TOTAL DE 210 REVISIONES ERRONEAS EN ESTA RUTA. AL INCREMENTAR A 11 EL NUMERO DE PUNTOS, ESTO NOS DA 20 POSIBLES POSICIONES A REVISAR, LO CUAL ES IGUAL A 220. LLEGANDO A 15, TENEMOS 16 POSICIONES POSIBLES, LO CUAL ES IGUAL A 240. PERO CUANDO SE LLEGA A 16, EL NUMERO DE POSIBLES POSICIONES YA ES DE 15, LO CUAL DA 240. Y A PARTIR DE ESTE MOMENTO SE EMPIEZAN A DECREMENTAR. SI LLEGAMOS AL 25, EL NUMERO DE POSICIONES A REVISAR ES DE 6, LO CUAL NOS DA UN TOTAL DE 150.

COMO VEMOS, LA RAPIDEZ CON QUE EL SIMULADOR RESPONDA DEPENDE, DE QUE NUMERO DE RUTA SE ENCUENTRE, Y DE LA MAGNITUD QUE TENGA ESTA RUTA CON RESPECTO A LAS RUTAS ANTERIORES.

EL IDEAL DE RAPIDEZ ES CUANDO SE ENCUENTRA CON LA PRIMERA RUTA.

DESDE LUEGO EL ENCUENTRO CON RUTAS CONOCIDAS ES TOTALMENTE FORTUITO, Y LA SITUACION EN QUE SE ENCUENTREN TAMBIEN LO ES. Y ES QUE HAY LA POSIBILIDAD DE QUE EL SIMULADOR SE ENCUENTRE UNA RUTA; COMO AL PRIMER PUNTO ES MUY DIFICIL QUE LA RECONOSCA, HAY LA POSIBILIDAD DE QUE LA CONFUNDA CON OTRA, O QUE SIMPLEMENTE NO ENCUENTRE NINGUN PUNTO PARECIDO.

ENTONCES LA DECISION QUE TOME PODRIA LLEVARLO A RECORRER LA RUTA EN SENTIDO CONTRARIO, Y NO ESTA EN POSIBILIDAD DE RECONOCER LAS RUTAS CUANDO SE ENCUENTRA EN ESTA SITUACION. ESTE ES TAL VEZ SU PRINCIPAL DEFECTO, PERO TENIENDO EN CUENTA QUE EL SIMULADOR NO TIENE MUCHOS MEDIOS PARA HACERLO, NO PARECE TAN GRANDE.

DESDE LUEGO QUE HACER UN SISTEMA QUE RECONOSCA RUTAS EN SENTIDO CONTRARIO NO DEBE SER COSA DEL OTRO MUNDO, PUES ESTE MISMO SISTEMA SERVIRIA CON LA CONDICION DE QUE SE MODIFICARA SOLO LA RUTINA DE BUSQUEDA EN MEMORIA, SIN TENER QUE TOCAR PARA NADA OTRAS RUTINAS YA EXISTENTES.

ESTA ES UNA DE LAS VENTAJAS QUE PUEDE TENER ESTE SISTEMA, Y ES EL HECHO DE QUE SE PUEDEN HACER LAS MODIFICACIONES QUE SE DESEEN A CUALQUIERA DE LAS RUTINAS, SIN PELIGRO DE MODIFICAR LAS DEMAS. DESDE LUEGO PARA HACER MODIFICACIONES SE DEBE TENER EN CUENTA LA FORMA EN QUE LA INFORMACION ENTRA A LAS RUTINAS, Y LA FORMA EN QUE SALE DE LAS MISMAS. RESPETANDO ESTE ULTIMO PUNTO NO HAY LIMITE EN CUANTO A LAS MODIFICACIONES POSIBLES.

10.- CONCLUSIONES Y POSIBLES APLICACIONES

UNA VEZ QUE YA SE TIENE UN ALGORITMO QUE RESUELVE ADECUADAMENTE CUALQUIER LABERINTO, SE PUEDE EMPEZAR A PENSAR EN SIMPLIFICAR EL NIVEL DE PROGRAMACION PARA PODER ADAPTARLO A UN PROCESADOR DE DIMENSIONES MENORES. ES POSIBLE ASI MISMO LLEGAR A IMPLEMENTAR EL SISTEMA EN UN MICROPROCESADOR, LO CUAL SERIA LA META IDEAL, PUES SI SE LOGRA ESTO, LA SOLUCION ESTARIA EN UNOS CUANTOS CIRCUITOS INTEGRADOS, DE MANERA QUE FUERA VIABLE MONTAR ESTE CIRCUITO EN UN DISPOSITIVO ELECTROMECANICO CAPAZ DE RESOLVER FISICAMENTE CUALQUIER LABERINTO.

TAL VEZ LOS PROBLEMAS MAYORES QUE SE ENFRENTARIAN EN ESE CASO, SERIAN DE INDOLE MECANICO, MAS QUE ELECTRONICO, AUNQUE DESDE LUEGO MUCHOS DE ESTOS PROBLEMA SE PODRIAN RESOLVER CON LA AYUDA DEL MISMO MICROPROCESADOR, DE MANERA QUE SE TENDRIAN QUE HACER OTROS PROGRAMAS DE SOPORTE.

LOGRAR ESTE OBJETIVO VUELVE A SER SIMPLEMENTE UN CAMBIO DE UNA DE LAS RUTINAS DEL PROGRAMA, QUE ES LA DE LA INTERFASE, EN LA CUAL SE TENDRIA QUE SUSTITUIR ESTA POR TODO UN PROYECTO DE COMUNICACION CON EL MEDIO EXTERIOR.

CREO QUE LA PRINCIPAL APLICACION QUE PODRIA TENER ESTE SIMULADOR ES EN EL CAMPO DE LA PEDAGOGIA. SUS ALCANCES VAN DESDE SER UN EJEMPLO EXCELENTE PARA LOS ESTUDIANTES DE INGENIERIA CIBERNETICA, PUDIENDO SER UN RETO PARA ELLOS MEJORAR EL SISTEMA. PARA LOS ALUMNOS DE INGENIERIA ELECTRONICA, ES EVIDENTE QUE SERIA UNA TENTACION CONVERTIR ESTE SIMULADOR EN UN ENTE FISICO, PARTE EN LA CUAL TENDRIAN MUCHO QUE HACER LOS ESTUDIANTES DE INGENIERIA MECANICA.

OTRA DE LAS APLICACIONES QUE PODRIA TENER UN SISTEMA DE ESTE TIPO ES LA DE DARNOS UNA IDEA DE COMO CREAR MAQUINAS CAPACES DE APRENDER CASI SOLAS, PERO LO MAS IMPORTANTE ES QUE ESTAS MAQUINAS PUEDAN SER ADIESTRADAS POR PERSONAS QUE NO NECESITEN SABER NADA DE PROGRAMACION. ES DECIR, PERSONAS QUE CONOSCAN UNA TAREA, QUE SEAN EXCELENTES TECNICOS PERO QUE CARESCAN DE LOS CONOCIMIENTOS NECESARIOS PARA SOPORTAR EL CHOQUE CULTURAL QUE PUEDE SIGNIFICAR EL ENFRENTARSE A UNA COMPUTADORA.

PIENSO QUE ESTE ES UN EJEMPLO QUE APENAS RAYA EN LOS LINDEROS DE LA INTELIGENCIA ARTIFICIAL, SIN EMBARGO TAMBIEN CREO QUE CUANDO UNO QUIERE ADENTRARSE EN UN CAMPO NUEVO, TOTALMENTE DESCONOCIDO, DEBE PROBAR DE ALGUNA MANERA HASTA DONDE LLEGAN SUS CONOCIMIENTOS EN LAS AREAS ALLEGADAS AL TEMA. CREO QUE FUE UNA EXCELENTE GIMNASIA MENTAL, Y UNA PRUEBA DE QUE LOS PROBLEMAS MAS SIMPLES NO LO SON SINO HASTA QUE SE HACE UN ANALISIS DETALLADO DE LOS MISMOS.

CADA PASO QUE DABA, ME ENCONTRABA CON COSAS NUEVAS EN LAS CUALES NO HABIA PENSADO ANTES. CADA SITUACION REQUERIA UNA SOLUCION, LA CUAL SE ENCONTRO LLEVANDO UN REGISTRO ORDENADO DE TODAS LAS IDEAS QUE SURGIERON EN EL CAMINO. ASI PUES LLEGO EL MOMENTO EN QUE ESTUVE SEGURO DE QUE EL ANALISIS HABIA LLEGADO AL FINAL. UNA VEZ DADO ESTE PASO, TODOS LOS SIGUIENTES NO PRESENTARON PROBLEMAS MAYORES.

ESTOY CONVENCIDO DE QUE ES UN PROYECTO DEMASIADO GRANDE PARA UNA SOLA PERSONA. PERO EL PRIMER PASO ESTA DADO. SE INTENTO LOGRAR EL MODELO DE UNA SOLUCION; SE LOGRO DE LA MANERA MAS SATISFACTORIA.

BIBLIOGRAFIA

* HAMILTON Pamela
"Just a phone call will transfer funds"
Electronics
United States of America
Vol. 54 No. 2
pp. 53-55
27/I/1981

* GARGLIANO Tim A.
"Giving voice to text"
Electronics
United States of America
Vol. 54 No. 3
pp. 117-125
10/II/1981

* SMITH Kevin
"Unit identifies words in run-on speech"
Electronics
United States of America
Vol. 54 No. 4
pp. 82
24/II/81

* AHRENS Paul
"Speech chip timeshares a 2-pole section to create a 12-pole filter"
Electronics
United States of America
Vol. 54 No. 5
pp. 177-180
10/III/1981

* COSTLOW Terry
"Speech recognition"
Electronics
United States of America
Vol. 54 No. 6
pp. 42
24/III/1981

* GLUSKOFF Douglas
"Portable Scanner reads handwritten letters and figures"
Electronics
United States of America
Vol. 54 No. 7
pp. 81-82
7/IV/1981

* LINEBACK Robert J.
"Voice messaging a future giant"
Electronics
United States of America
Vol. 54 No. 8
pp. 99-100
21/IV/1981

* NEFF Robert
"Personal computer accept oral input"
Electronics
United States of America
Vol 54 No. 12
pp. 78-80
16/VI/1981

* VACCA Anthony A.
"Supercomputer outdoes itself by designing its successor"
Electronics
United States of America
Vol. 54 No. 13
pp. 106-110
30/VI/1981

* LINEBACK Robert J.
"Development systems add in-house"
Electronics
United States of America
Vol. 54 No. 23
pp. 38-39
17/XI/1981

* NEFF Robert
"Japanese welcome voice recognition"
Electronics
United States of America
Vol. 55 No. 3
pp. 97-98
10/II/1982

* IVERSON Wesley R.
"Vision systems gain smarts"
Electronics
United States of America
Vol. 55 No. 7
pp. 89-90
7/IV/1982

* PRESSER-CARDENAS Y MARIN
Ciencias de la Computacion
MEXICO D.F.
Ed. Limusa
pp. 260-308
Vol. II
1980