```cpp
#include <iostream>
#include <stdio.h>
#include <windows.h>
#include <string>
#include <vector>
#include <fstream>
#include<math.h>
#define D cout<<"DEBUG"<<endl;
#include <unistd.h>
#include<ctime>
using namespace std;
int temp;
string str1;
ofstream maparray;
ifstream file;
char tmp_map[18][32];
char *argv[5];
char map[18][32] = {
	"+#############################+",
	"|                             |",
	"|                             |",
	"|## ########### ##   #########|",
	"|   |                         |",
	"| | |### |  |           |     |",
	"| |      |  | |###  |   |  |  |",
	"| | #####|  | |      ## |     |",
	"| |           |###  |      |  |",
	"| |##### ###         ##       |",
	"|          ######  ####### ###|",
	"|       .    .               .|",
	"|# ### ####      ###   #### # |",
	"|             .               |",
	"|                             |",
	"|             .               |",
	"|                             |",
	"+#############################+"
	};


void ShowMap()
{
	for(int i = 0; i < 18; i++) {
	//	printf("%s\n",map[i] );
		cout<<map[i]<<endl;
	}
}

void gotoxy( short x, short y )
{

    HANDLE hStdout = GetStdHandle(STD_OUTPUT_HANDLE) ;
    COORD position = { x, y } ;

    SetConsoleCursorPosition( hStdout, position ) ;

}

class entity {
public:
	entity( int x, int y ){
		this ->x = x;
		this ->y = y;
	}

	void move_x( int p ){
		if( map[y][x+p] == ' ' ) x += p;
	}

	void move_y( int p ){
		if( map[y+p][x] == ' ' ) y += p;
	}

	void move( int p, int q ){
		x += p;
		y += q;
	}

	int get_x(){ return x; }
	int get_y(){ return y; }

	void draw( char p ){
		map[x][y] = p;
		gotoxy( x, y ); printf( "%c", p );
		}

private:
		int x;
		int y;
};

struct walk {
		short walk_count;
		short x;
		short y;
		short back;
};

struct target {
		short x;
		short y;
};

vector<target> walk_queue;

vector<walk> BFSArray;

void AddArray( int x, int y, int wc , int back ){
		if( tmp_map[y][x] == ' ' || tmp_map[y][x] == '.' ){
				tmp_map[y][x] = '#';
				walk tmp;
				tmp.x = x;
				tmp.y = y;
				tmp.walk_count = wc;
				tmp.back = back;
				BFSArray.push_back( tmp );
		}
}

void FindPath( int sx, int sy, int x, int y ){
		memcpy( tmp_map, map, sizeof(map) );
		BFSArray.clear();
		walk tmp;
		tmp.x = sx;
		tmp.y = sy;
		tmp.walk_count = 0;
		tmp.back = -1;
		BFSArray.push_back( tmp );

		int i = 0;
		while( i < BFSArray.size() ){
				if( BFSArray[i].x == x && BFSArray[i].y == y ){
						walk_queue.clear();
						target tmp2;
						while( BFSArray[i].walk_count != 0 ){
								tmp2.x = BFSArray[i].x;
								tmp2.y = BFSArray[i].y;
								walk_queue.push_back( tmp2 );

								i = BFSArray[i].back;
						}

						break;
				}

				AddArray( BFSArray[i].x+1, BFSArray[i].y, BFSArray[i].walk_count+1, i );
				AddArray( BFSArray[i].x-1, BFSArray[i].y, BFSArray[i].walk_count+1, i );
				AddArray( BFSArray[i].x, BFSArray[i].y+1, BFSArray[i].walk_count+1, i );
				AddArray( BFSArray[i].x, BFSArray[i].y-1, BFSArray[i].walk_count+1, i );

				i++;
		}

		BFSArray.clear();
}
int GetDirectionRight(int tick,ifstream& file){

file.open("./check/"+(to_string(temp)+".GEN"));
string line;

  if (file.is_open())
  {

    getline (file,line) ;


    file.close();
  }

  if(line[0]=='R'){

                return 1;

  }

  return 0;
}
int GetDirectionLeft(int tick){
file.open("./check/"+(to_string(temp)+".GEN"));
string line;

  if (file.is_open())
  {

                getline (file,line) ;

    file.close();
  }

  if(line[0]=='L'){

                return 1;

  }

  return 0;
}

int GetDirectionUp(int tick){
file.open("./check/"+(to_string(temp)+".GEN"));
string line;

  if (file.is_open())
  {

    getline (file,line) ;


    file.close();
  }

  if(line[0]=='U'){

                return 1;

  }

  return 0;
}

int GetDirectionDown(int tick){
file.open("./check/"+(to_string(temp)+".GEN"));
string line;

  if (file.is_open())
  {

     getline (file,line) ;

    file.close();
  }

  if(line[0]=='D'){

                return 1;

  }

  return 0;
}

writePos(int x,int y,int ex, int ey){
                maparray<<x<<";"<<y<<";"<<ex<<";"<<ey<<";";


}

int calculateUp(int x,int y){
for(int i=y;i>0;i--){
		if(map[i][x]==0x23 || map[i][x]==0x7c || map[i][x]==0x45 || map[i][x]==0x2e){
		maparray<<y-i<<";"<<(int)map[i][x]<<";";

		break;
		}
		}
}

int calculateRight(int x,int y){

for(int i=x;i<32;i++){

		if(map[y][i]==0x7c || map[y][i]==0x23 || map[y][i]==0x45 || map[y][i]==0x2e ){


		maparray<<i-x<<";"<<(int)map[y][i]<<";";

		break;
		}

}
}

int calculateDown(int x,int y){
for(int i=y;i<18;i++){

		if(map[i][x]==0x7c || map[i][x]==0x23 ||map[i][x]==0x45 || map[i][x]==0x2e){
		maparray<<i-y<<";"<<(int)map[i][x]<<";";

		break;
		}

}


}


int calculateLeft(int x,int y){
for (int i=x;i>=0;i--){

		if(map[y][i]==0x7c || map[y][i]==0x23 || map[y][i]==0x45 || map[y][i]==0x2e){
		maparray<<x-i<<";"<<(int)map[y][i]<<";";

		break;
		}

}


}



int main(int argc,char *argv[])
{


		int tick;

		temp=strtol(argv[1],NULL,10);
		maparray.open("./check/"+(to_string(temp)+".map"));

		if(!maparray){
				cout<<"ERROR OPENING MAP FILE";
		}

		file.open("./check/"+(to_string(temp)+".GEN"));


    bool running = true;
		int x = 15; // hero x
		int y = 16; // hero y
		int old_x;
		int old_y;
		int ex = 1;
		int ey = 1;
		int pts = 0;
		char diffi='N';
		int speedmod = 3;
		diffi='N';
		if( diffi == 'N' ){
				speedmod = 2;
    }else if( diffi == 'H' ){
        speedmod = 1;
    }

    system("cls");
 ShowMap();

    gotoxy( x, y ); cout << "H";

    int frame = 0;
clock_t begin = clock();
    FindPath( ex,ey,x,y );

    while( running ){
        tick++;
        if(tick>100){
            break;
        }

        maparray.open("./check/"+(to_string(temp)+".map"));
        gotoxy( x, y ); cout << " ";

        old_x = x;
        old_y = y;


    if( ex == x && ey == y ){
            break;
        }


        if ( GetDirectionUp(tick) ){

                if( ex == x && ey == y-1 ){
            break;
        }

            if( map[y-1][x] == '.' ){ y--; pts+=60; } else
            if( map[y-1][x] == ' ' ) y--;
        }
        if ( GetDirectionDown(tick) ){
                if( ex == x && ey == y+1 ){
            break;
        }
            if( map[y+1][x] == '.' ){ y++; pts+=60; } else
            if( map[y+1][x] == ' ' ) y++;
        }
        if ( GetDirectionLeft(tick) ){
                if( ex == x-1 && ey == y ){
            break;
        }
            if( map[y][x-1] == '.' ){ x--; pts+=60; } else
            if( map[y][x-1] == ' ' ) x--;
        }
        if ( GetDirectionRight(tick,file) ){
                if( ex == x+1 && ey == y ){
            break;
        }
            if( map[y][x+1] == '.' ){ x++; pts+=60; } else
            if( map[y][x+1] == ' ' ) x++;
        }



        if( old_x != x || old_y != y ){
            FindPath( ex,ey,x,y );
        }

        gotoxy( x,y ); cout << "H";

        map[ey][ex] = '.';
        gotoxy( ex, ey ); cout << ".";

        if( frame%speedmod == 0 && walk_queue.size() != 0 ){
            ex = walk_queue.back().x;
            ey = walk_queue.back().y;
            walk_queue.pop_back();
        }

        gotoxy( ex, ey ); cout << "E";

        if( ex == x && ey == y ){
            break;
```

```
			}


			gotoxy( 32, 18 );
			gotoxy( 32, 1 ); cout << pts;

		//	sleep(1);
			usleep(50000)		;

			cout<<endl;
			calculateLeft(x,y);
			calculateUp(x,y);
			calculateRight(x,y);
			calculateDown(x,y);
			writePos(x,y,ex,ey);


			maparray.flush();
			maparray.close();


			if(frame==0){
				long tempargv1=strtol(argv[1],NULL,10);
				str1="gencheck.exe ";
				str1+=to_string(tempargv1);


				string st1="./check/";
				st1+=to_string(tempargv1);
				st1+=".best";




			if(ifstream(st1)){

			str1+=" 1";
			}else {

			str1+=" 1";

		}

			}
			if(frame>0){
			system(str1.c_str());
		}
			frame++;
		}

	clock_t end = clock();
	double elapsed_secs = double(end - begin) / CLOCKS_PER_SEC;

		system("cls");


		ofstream retu("./check/"+to_string(temp)+".return");
		if(!retu){
			cout<<".return-file "<<retu<<" not opened";
		}

		pts+=tick;
		retu<<pts;
		retu.flush();
		retu.close();
		printf("You Lose and your score is : %i", pts );
		string del="del ";
		del+=".\\check\\";
		del+=to_string(temp);
		del+=".map";
		maparray.close();
		system(del.c_str());
		system("pause");
		return 0;
	}
```