Vinnitsa IT Academy



Theme of project: « Ant in labyrinth»

[v.1.0.0.]

Realized by: Vadim Malushko

B-15 yellow

Review: Olekcii Tiurenkov

Vinnitsa 2016

1. System requirements

2.About project

3. User guide

4.Programmer instruction

5.About author

**1. System requirements**

System requirements include:

Intel 80486DX2/66 MHz or a compatible CPU with a math coprocessor, (Pentium processor recommended), 16 MB or RAM(24 recommended). At least 500 MB of space available on HDD.

Upgrading from Windows 7 . VGA or higher resolution monitor (640×480). CD-ROM or DVD-ROM or flash drive.

**2.About project**

“Ant in labyrinth” developed on C language and use in console. In the program “Ant” it is symbolic name of symbol where he must find a way to exit the labyrinth. Ant also must be trained and pass labyrinth faster.

**3. User guide**

1. This program consists of one operation. Enter the file “ant.exe” and watch how the “Ant” remambar its way. That’s all.

**4.Programmer instruction**

One of the major advantages of Qt Creator is that allows a team developers to share a project across different development platforms (Microsoft Windows, Mac OS X, and Linux) with a common tool for development and debugging.

**Element of code**

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#include <windows.h>

#include <antlibery.h>

#include <conio.h>

#include <unistd.h>

#include <time.h>

#define TRUE 1

#define FALS 0

#define hCon GetStdHandle(STD\_OUTPUT\_HANDLE)

typedef struct { //Create structure of (int x, int y) coordinates and console position(COORD new\_Position);

COORD new\_Position; // standard struct for console in ,,C’’ language.

int x;

int y;

}NextAnt;

struct ListCordinats{//Structure for stack;

unsigned count;

unsigned X\_position;

unsigned Y\_position;

struct ListCordinats \*next,\*prev; //Create pointer ‘next’ and ‘previous’ for addressable location

};

typedef struct ListCordinats \*PNode;

typedef struct {//Structure which consist head and tail of this stack;

PNode Head;

PNode Tail;

}Stack;

void PUSH(unsigned count,unsigned X, unsigned Y, Stack \*p){//function for insert new element

PNode NewNode;

NewNode = (PNode)malloc(sizeof(struct ListCordinats));

NewNode->count=count;

NewNode->X\_position = X;

NewNode->Y\_position = Y;

NewNode->next = p->Head;

NewNode->prev = NULL;

if(p->Head)

p->Head->prev = NewNode;

p->Head = NewNode;

if(!p->Tail) p->Tail = p->Head;

}

void POP\_element(Stack \*p){//deleting element in stack

p->Head= p->Head->next;

}

int Serch\_in\_Stack(NextAnt \*x,Stack \*p, PNode Tail, int count){//inside function for data searching in stack

PNode Tamp;

Tamp = Tail;

if(count!=1){

while(Tamp!=NULL){//circle for running of stack

if((x->x==Tamp->X\_position)&&(x->y==Tamp->Y\_position)&&Tamp->count==count-1){

return TRUE; //if find same coordinates and if count not null – return “TRUE”

}

else if ((x->x==Tamp->X\_position)&&(x->y==Tamp->Y\_position)){

PUSH(count,x->x,x->y,p);

move\_ant\_for\_Stack(x,Tail,Tamp);

return 0;

}

Tamp=Tamp->prev;

}

}

return FALS;

}

int Search\_in\_function\_move\_ant(NextAnt\*x, PNode Tail, PNode Tamp){ //searching in stack in ant memory

PNode BoofTemp;

BoofTemp = Tail;

while(BoofTemp!=NULL){

if((x->x==BoofTemp->X\_position)&&(x->y==BoofTemp->Y\_position)){

Tamp=BoofTemp;

break;

}

BoofTemp=BoofTemp->prev;

}

}

void move\_ant\_for\_Stack(NextAnt\*x, PNode Tail, PNode Tamp){ // drawing symbol “Ant” by coordinates in stack

while(!(MAP[x->y][x->x]==3)){

x->x=x->new\_Position.X = Tamp->X\_position;

x->y=x->new\_Position.Y = Tamp->Y\_position;

Tamp=Serch\_in\_function\_move\_ant(x,Tail,Tamp);

draw\_ant(x);

Sleep(30);

Tamp=Tamp->prev;

if(Tamp==NULL){

Tamp=Tail;

}

clear\_ant(x);

}

}

void display(){//screen size

system("mode con cols=540 lines=500");

}

void func\_labyrinth (const int map[41][41]){//draw labyrinth by detention in array of meanings (0,1 or 3) who are located in antlibery.h file

int x,y;

for(x=0;x<41;x++){

for(y=0;y<41;y++){

switch (map[x][y]){

case 0:

printf("%c", 32);

break;

case 1:

printf("%c", 178);

break;

case 3:

printf("%c", 32);

break;

}

}

printf("\n");

}

}

void Randfunction(NextAnt \*p, const int map[41][41]){// creates random new coordinates

int x,y;

time\_t t;

unsigned s = 2;

srand((unsigned) time(&t));

while(s!=0){

p->x = rand()%41;

p->y = rand()%41;

x = p->x;

y = p->y;

s = map[y][x];

}

}

void move\_Ant\_LEFT(int count,NextAnt \*x,Stack \*p){//draw symbol on the left side of a current position and delete symbol from a current position

int TRUE\_or\_FALSE = 0;

while(!(MAP[x->y][x->x]==3)){

clear\_ant(x);

x->new\_Position.X -= 1;

x->x-=1;

TRUE\_or\_FALSE = Serch\_in\_Stack(x,p,p->Tail,count);

if(TRUE\_or\_FALSE==TRUE){

POP\_element(p);

count--;

}

else if(TRUE\_or\_FALSE==FALSE){

count++;

PUSH(count,x->x, x->y,p);

}

draw\_ant(x);

Sleep(30);

if(MAP[x->y+1][x->x]==0){

move\_Ant\_DOWN(count,x,p);

}

else if(MAP[x->y][x->x-1]==1&&MAP[x->y-1][x->x]==0){

move\_Ant\_UP(count,x,p);

}

else if(MAP[x->y][x->x-1]==1){

move\_Ant\_RIGHT(count,x,p);

}

}

}

void move\_Ant\_RIGHT(int count,NextAnt \*x,Stack \*p){ //draw symbol on the right side of a current position and delete symbol from a current position

int TRUE\_or\_FALSE = 0;

while(!(MAP[x->y][x->x]==3)){

clear\_ant(x);

x->new\_Position.X += 1;

x->x+=1;

TRUE\_or\_FALSE = Serch\_in\_Stack(x,p,p->Tail,count);

if(TRUE\_or\_FALSE==TRUE){

POP\_element(p);

count--;

}

else if(TRUE\_or\_FALSE==FALSE){

count++;

PUSH(count,x->x, x->y,p);

}

draw\_ant(x);

Sleep(30);

if(MAP[x->y-1][x->x]==0){

move\_Ant\_UP(count,x,p);

}

else if(MAP[x->y][x->x+1]==1&&MAP[x->y+1][x->x]==0){

move\_Ant\_DOWN(count,x,p);

}

else if(MAP[x->y][x->x+1]==1){

move\_Ant\_LEFT(count,x,p);

}

}

}

void move\_Ant\_UP(int count,NextAnt \*x,Stack \*p){ //draw symbol on the up of a current position and delete symbol from a current position

int TRUE\_or\_FALSE = 0;

while(!(MAP[x->y][x->x]==3)){

clear\_ant(x);

x->new\_Position.Y-= 1;

x->y-=1;

TRUE\_or\_FALSE = Serch\_in\_Stack(x,p,p->Tail,count);

if(TRUE\_or\_FALSE==TRUE){

POP\_element(p);

count--;

}

else if(TRUE\_or\_FALSE==FALSE){

count++;

PUSH(count,x->x,x->y,p);

}

draw\_ant(x);

Sleep(30);

if(MAP[x->y][x->x-1]==0){

move\_Ant\_LEFT(count,x,p);

}

else if(MAP[x->y-1][x->x]==1&&MAP[x->y][x->x+1]==0){

move\_Ant\_RIGHT(count,x,p);

}

else if(MAP[x->y-1][x->x]==1){

move\_Ant\_DOWN(count,x,p);

}

}

}

void move\_Ant\_DOWN(int count,NextAnt \*x,Stack \*p){ //draw symbol on the down of a current position and delete symbol from a current position

int TRUE\_or\_FALSE = 0;

while(!(MAP[x->y][x->x]==3)){

clear\_ant(x);

x->new\_Position.Y+= 1;

x->y+=1;

TRUE\_or\_FALSE = Serch\_in\_Stack(x,p,p->Tail,count);

if(TRUE\_or\_FALSE==TRUE){

POP\_element(p);

count--;

}

else if(TRUE\_or\_FALSE==FALSE){

count++;

PUSH(count,x->x, x->y,p);

}

draw\_ant(x);

Sleep(30);

if(MAP[x->y][x->x+1]==0){

move\_Ant\_RIGHT(count,x,p);

}

else if(MAP[x->y+1][x->x]==1&&MAP[x->y][x->x-1]==0){

move\_Ant\_LEFT(count,x,p);

}

else if(MAP[x->y+1][x->x]==1){

move\_Ant\_UP(count,x,p);

}

}

}

void draw\_ant(NextAnt \*nextAnt){ //draw symbol by coordinates

SetConsoleCursorPosition(hCon,nextAnt->new\_Position);

printf("%c", 15);

}

void clear\_ant(NextAnt \*nextAnt){//delete symbol by coordinates

SetConsoleCursorPosition(hCon,nextAnt->new\_Position);

printf("%c", 0);

}

int main()

{

display();

NextAnt nextAnt;

Stack stack;

stack.Head = NULL;

stack.Tail = NULL;

unsigned count = 0;

func\_labyrinth(MAP);

while (1){//infinity circle

clear\_ant(&nextAnt);

Randfunction(&nextAnt, MAP);

nextAnt.new\_Position.X = nextAnt.x;

nextAnt.new\_Position.Y = nextAnt.y;

if(MAP[nextAnt.y-1][nextAnt.x]==0){

move\_Ant\_UP(count,&nextAnt,&stack);

}

else if(MAP[nextAnt.y][nextAnt.x+1]==0){

move\_Ant\_RIGHT(count,&nextAnt,&stack);

}

else if(MAP[nextAnt.y+1][nextAnt.x]==0){

move\_Ant\_DOWN(count,&nextAnt,&stack);

}

else if(MAP[nextAnt.y][nextAnt.x-1]==0){

move\_Ant\_LEFT(count,&nextAnt,&stack);

}

}

return 0;

}

**5.About author**

This project was realized by Vadim Malushko, B15-yellow 2016.