Snake Game

Adrian Dobosz

RPN8RH

List of functions:

# getch()

The function getch() is a non-standard function. It is declared in “conio.h” header file. Mostly it is used by Turbo C. It is not a part of C standard library. It immediately returns the entered character without even waiting for the enter key.

# system()

The C library function int system(const char \*command) passes the command name or program name specified by command to the host environment to be executed by the command processor and returns after the command has been completed.

# FILE

declaration for communication between the file and program.

# fopen()

The C library function FILE \*fopen(const char \*filename, const char \*mode) opens the filename pointed to, by filename using the given mode.

filename − This is the C string containing the name of the file to be opened.

mode − This is the C string containing a file access mode. It includes

# fprintf()

The C library function int fprintf(FILE \*stream, const char \*format, ...)sends formatted output to a stream.

stream − This is the pointer to a FILE object that identifies the stream.

format − This is the C string that contains the text to be written to the stream.

# fscanf()

The C library function int fscanf(FILE \*stream, const char \*format, ...)reads formatted input from a stream.

stream − This is the pointer to a FILE object that identifies the stream.

format − This is the C string that contains one or more of the following items

# fclose()

The C library function int fclose(FILE \*stream) closes the stream. All buffers are flushed.

# strcpy()

The C library function char \*strcpy(char \*dest, const char \*src) copies the string pointed to, by src to dest.

# switch()

A switch statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each switch case.

The syntax for a switch statement in C programming language is as follows −

switch(expression) {  
  
 case constant-expression :  
 statement(s);  
 break; /\* optional \*/  
   
 case constant-expression :  
 statement(s);  
 break; /\* optional \*/  
   
 /\* you can have any number of case statements \*/  
 default : /\* Optional \*/  
 statement(s);  
}

The following rules apply to a switch statement −

-The expression used in a switch statement must have an integral or enumerated type, or be of a class type in which the class has a single conversion function to an integral or enumerated type.

-You can have any number of case statements within a switch. Each case is followed by the value to be compared to and a colon.

-The constant-expression for a case must be the same data type as the variable in the switch, and it must be a constant or a literal.

-When the variable being switched on is equal to a case, the statements following that case will execute until a break statement is reached.

-When a break statement is reached, the switch terminates, and the flow of control jumps to the next line following the switch statement.

-Not every case needs to contain a break. If no break appears, the flow of control will fall through to subsequent cases until a break is reached.

-A switch statement can have an optional default case, which must appear at the end of the switch. The default case can be used for performing a task when none of the cases is true. -No break is needed in the default case.

# malloc()

The C library function void \*malloc(size\_t size) allocates the requested memory and returns a pointer to it.

# realloc()

The C library function void \*realloc(void \*ptr, size\_t size) attempts to resize the memory block pointed to by ptr that was previously allocated with a call to malloc or calloc.

My solution:

# Print board

int map\_size=15;

int map[map\_size\*map\_size];

mapone(map\_size,map);

while(1){

system("CLS");

mapzero(map\_size,map);

printmap(map\_size,map);

}

void mapzero(int map\_size,int \*map)

{

for(int i=1;i<map\_size-1;i++)

{

for(int j=1;j<map\_size-1;j++)

{

\*(map+i\*map\_size+j)=0;

}

}

}

void mapone(int map\_size,int \*map)

{

for(int i=0;i<map\_size;i++)

{

for(int j=0;j<map\_size;j++)

{

\*(map+i\*map\_size+j)=1;

}

}

}

void printmap(int map\_size,int \*map)

{

for(int i=0;i<map\_size;i++)

{

for(int j=0;j<map\_size;j++)

{

printf("%d ",\*(map+i\*map\_size+j));

}

printf("\n");

}

}

# print head on board

typedef struct

{

int x;

int y;

} snake;

snake pozition={rand()%(map\_size-2)+1,rand()%(map\_size-2)+1};

void snake\_map(snake p,int \*map,int x)

{

\*(map+p.y\*x+p.x)=1;

}

# move head

snake move(snake p,int key)

{

switch(key)

{

case'w': p.y--; break;

case's': p.y++; break;

case'a': p.x--; break;

case'd': p.x++; break;

case'W': p.y--; break;

case'S': p.y++; break;

case'A': p.x--; break;

case'D': p.x++; break;

default: break;

}

return p;

}

int key=0;

while(1)

{

key=getch();

pozition=move(pozition,key);

}

# add “game over” situation

void game\_over(int map\_size,int \*map)

{

for(int i=0;i<map\_size;i++)

{

for(int j=0;j<map\_size;j++)

{

printf("%d ",\*(map+i\*map\_size+j));

if(i==map\_size/2 && j==map\_size/2-3)

{

printf("GAME OVER ");

j+=5;

}

}

printf("\n");

}

}

if(pozition.x==0 || pozition.x==map\_size-1 || pozition.y==map\_size-1 || pozition.y==0)

{

game\_over(map\_size,map);

Sleep(2000);

return 0;

}

# add randomly spawning fruit

snake fruit={rand()%(map\_size-2)+1,rand()%(map\_size-2)+1};

if(fruit.x==pozition.x && fruit.y==pozition.y)

{

fruit.x=rand()%(map\_size-2)+1;

fruit.y=rand()%(map\_size-2)+1;

apple\_map(fruit,map,map\_size);

}

# snake tail

snake \*tail;

int lenght=0;

tail=malloc((lenght+2) \* sizeof (snake ));

for(int i=lenght;i>=0;i--)

{

tail[i+1]=tail[i];

tail[0]=pozition;

}

if(fruit.x==pozition.x && fruit.y==pozition.y)

{

lenght++;

realloc(tail,(lenght+2) \* sizeof (snake ));

}

for(int i=lenght;i>=1;i--)

{

if(tail[i].x==pozition.x && tail[i].y==pozition.y){

game\_over(map\_size,map);

return 0;

}

# score and score saving

int score=0;

if(fruit.x==pozition.x && fruit.y==pozition.y)

{

score+=10;

}

printf("score: %d",score);

for(int i=lenght;i>=1;i--)

{

if(tail[i].x==pozition.x && tail[i].y==pozition.y){

game\_over(map\_size,map);

Sleep(2000);

socres(score,score,1);

return 0;

}

}

if(pozition.x==0 || pozition.x==map\_size-1 || pozition.y==map\_size-1 || pozition.y==0)

{

game\_over(map\_size,map);

Sleep(2000);

socres(score,score,1);

return 0;

}

void socres(int s,int s1,int ile)

{

FILE \*fp;

fp=fopen("scoretabela.txt", "w");

if(fp==NULL)

{

system("cls");

printf("not working");

}

score\_name name, name1;

system("cls");

if (ile == 1)

{

name.score=s;

printf("Your score is: %d\nPlease write your name:",name.score);

scanf("%s",name.name);

fprintf (fp, "%s %d\n",name.name,name.score);

}

if (ile == 2)

{

name.score=s;

name1.score=s1;

printf("Player 1 score is: %d\nPlayer 2 score is: %d\nPlease write Player 1 name:",name.score,name1.score);

scanf("%s",name.name);

printf("Please write Player 2 name:");

scanf("%s",name1.name);

fprintf (fp, "%s %d\n", name.name,name.score);

}

fclose(fp);

}

# score soting

void scoreboard()

{

FILE \*sc = fopen("scoreboard.txt","r");

score\_name board[12];

int entryCount=0 ,i = 0 ,ch,temp;

char empo[15];

FILE \*input = fopen("scoretabela.txt", "r");

while ((ch = fgetc(input)) != EOF)

{

if (ch == '\n')

entryCount++;

}

for (; i < entryCount; i++)

{

fscanf(input, "%s%d", board[i].name,&board[i].score);

}

for (; i < 12; i++)

{

fscanf(input, "%s %d", board[i].name,&board[i].score);

}

fclose(sc);

for (i = 0; i < entryCount; i++)

{

for (int j = 0; j < entryCount; j++)

{

if (i == j) continue;

if (board[i].score < board[j].score)

{

temp = board[i].score;

board[i].score = board[j].score;

board[j].score = temp;

strcpy(empo, board[i].name);

strcpy(board[i].name, board[j].name);

strcpy(board[j].name, empo);

}

}

}

sc=fopen("scoreboard.txt","w");

for (i = 0; i < 10; i++)

{

fprintf(sc,"%s %d\n",board[i].name,board[i].score);

}

fclose(input);

fclose(sc);

}

Reference

www.tutorialspoint.com