



**ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ**

**ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ**

**DEPARTMENT of ELECTRICAL and COMPUTER ENGINEERING**

**ΟΜΑΔΑ GAME DEVELOPMENT**

**Το πρώτο μου παιχνίδι στην C**

**Κρυμμένος Θησαυρός**

**ΧΡΗΣΤΟΣ ΤΥΠΟΥ**

## **Σκοπός του παιχνιδιού**

Ο στόχος είναι να δημιουργήσουμε ένα παιχνίδι κρυμμένου θησαυρού, στο οποίο ο παίκτης θα πρέπει να βρει 20 θησαυρούς σε έναν πίνακα 23x79 θέσεων.

Υπάρχουν 30 ηλεκτροφόρα εμπόδια μέσα στον πίνακα και στα περιθώρια του πίνακα υπάρχει ηλεκτροφόρος φράκτης. Τα εμπόδια και τα περιθώρια είναι αδιαπέραστα, και ο τρόπος τοποθέτησης των εμποδίων θα πρέπει να είναι τέτοιος, ώστε πάντα να υπάρχει μια διαδρομή που θα επιτρέπει στον χρήστη να αναζητήσει τον θησαυρό.

Στόχος είναι, στο τέλος, να έχουμε κατανοήσει τα βασικά στοιχεία της γλώσσας C, αλλά και να έχουμε εμβαθύνει περισσότερο στη λειτουργία της εντολής rand() -> random, και να έχουμε αποκτήσει την πρώτη μας εμπειρία στο game development, το οποίο αποτελεί και το θέμα με το οποίο ασχολείται η ομάδα.

## **Η δεύτερη τροποποιημένη έκδοση με χρήση γραφικών**

### **Ανάλυση κώδικα στην c**

```
//εισαγάγουμε τις παρακάτω βιβλιοθήκες
```

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <conio.h>
```

```
#include <time.h>
```

```
//όρισμα σταθερών μεταβλητών
```

```
#define ROWS 23
```

```
#define COLS 79
```

```
#define A 5
```

```
#define B 30
```

```
#define C 20
```

```
//συνάρτηση η οποία τοποθετεί τα εμπόδια, τον χρήστη και τους θησαυρούς σε σωστά σημεία μέσα στον πίνακα
void empodia(int array_basic_operation[ROWS][COLS], int *p, int *o){

    int i,num_x,num_y,x=-1,y=-1;

    for(i=0; i<B; i++) {

        do{

            num_x = rand() % (ROWS+1);

            num_y = rand() % (COLS+1);

            }while((x==num_x || y==num_y) || num_x>21 || num_y>77 || abs(x-num_x)<2 || abs(y-num_y)<2 ||
num_x<=0 || num_y<=0 || array_basic_operation[num_x][num_y]==1 || array_basic_operation[num_x-
1][num_y-1]==1 || array_basic_operation[num_x+1][num_y+1]==1 || array_basic_operation[num_x+1][num_y-
1]==1 || array_basic_operation[num_x-1][num_y+1]==1 || array_basic_operation[num_x][num_y+1]==1 ||
array_basic_operation[num_x+1][num_y]==1 || array_basic_operation[num_x-1][num_y]==1 ||
array_basic_operation[num_x][num_y-1]==1);

            x = num_x;
```

```
y = num_y;  
array_basic_operation[num_x][num_y] = 1;  
}  
  
x=-1,y=-1;  
for(i=0; i<C; i++){  
    do{  
        num_x=rand()%ROWS+1;  
        num_y=rand()%COLS+1;  
        }while(array_basic_operation[num_x][num_y]==1 || abs(x-num_x-2)<2 || abs(y-num_y-2)<2 ||  
x==num_x || y==num_y || num_x>=22 || num_y<1 || num_y>=78 || num_x<1 || num_y<0 ||  
array_basic_operation[num_x][num_y]==2 || array_basic_operation[num_x-1][num_y-1]==2 ||  
array_basic_operation[num_x+1][num_y+1]==2 || array_basic_operation[num_x+1][num_y-1]==2 ||  
array_basic_operation[num_x-1][num_y+1]==2 || array_basic_operation[num_x][num_y+1]==2 ||  
array_basic_operation[num_x+1][num_y]==2 || array_basic_operation[num_x-1][num_y]==2 ||  
array_basic_operation[num_x][num_y-1]==2);
```

```
x=num_x;  
y=num_y;  
array_basic_operation[num_x][num_y]=2;  
}  
  
do{  
    num_x=rand()%(ROWS+1);  
    num_y=rand()%(COLS+1);  
}while(array_basic_operation[num_x][num_y]==2 || array_basic_operation[num_x][num_y]==1 ||  
num_x>22 || num_y>77 || num_x<=0 || num_y<=0);  
  
*p=num_x;  
*o=num_y;
```

```
array_basic_operation[num_x][num_y]='O';
}

//αρχικοποίηση του πίνακα να είναι μηδέν όλες οι θέσεις του αλλά και τοποθέτηση των περιθωρίων
void initializeBoard(int array_basic_operation[ROWS][COLS], int *z, int *c) {
    int i,j;

    for (i = 0; i < ROWS; i++) {
        for (j = 0; j < COLS; j++) {
            array_basic_operation[i][j]=0;
            if(j==78 || j==0 || i==0 || i==22){
                array_basic_operation[i][j]=1;
            }
        }
    }
}
```

```
//καλούμε την προηγούμενη συνάρτηση embodia για να τοποθετηθούν εμπόδια ο παιχτής και οι θησαυροί μέσα στον πίνακα  
embodia(array_basic_operation,z,c);  
}
```

```
//συνάρτηση εκτύπωσης του πίνακα στην οθόνη αφού πρώτα διαγράψουμε την προηγούμενη εγγραφή σε αυτήν  
void printBoard(int array_basic_operation[ROWS][COLS]) {
```

```
    //εντολή καθαρισμού του terminal
```

```
    system("cls");
```

```
    int i,j;
```

```
    for (i = 0; i < ROWS; i++) {
```

```
        for (j = 0; j < COLS; j++) {
```

```
            if(array_basic_operation[i][j]=='O'){


```

```
                printf("\U0001F600");//smile face emoji
```

```
 }else if(array_basic_operation[i][j]==0 || array_basic_operation[i][j]==2){  
    printf(" ■ ");  
 }else if(array_basic_operation[i][j]==3){  
    printf(" ■ ");  
 }else{  
    printf(" ✘ ");  
 }  
 printf("\n");  
 }  
 }  
  
//συνάρτηση η οποία εκτυπώνει πόσους θησαυρούς έχει βρει ο χρήστης  
void print_tropea(int array[C]){
```

```
int i;

for(i=0; i<C; i++){
    if(array[i]==1){
        printf("🏆 ");
    }
    printf("\n");
}

//συνάρτηση η οποία εκτυπώνει το πλήθος των ζωών που έχει ο χρήστης ακόμα (χάνει ζωή αν ακουμπήσει κάποιο εμπόδιο και παίρνει ζωή κάθε φορά που βρίσκει θησαυρό

void print_lifes_in_the_game(int array[A]){
    int i;
```

```
for(int i=0; i<A; i++){
    if(array[i]==1){
        printf(" ❤ ");
    }
    printf("\n");
}

//κύρια συνάρτηση του προγράμματος

int main() {
    int a=A;
    int life_in_the_game[A]={1,1,1,1,1};
    int found_tropea[C]={};
    int find=0;
    //εντολή αρχικοποίησης της γεννήτριας τυχαίων αριθμών
```

```
    srand(time(0));  
  
    int array_basic_operation[ROWS][COLS];  
  
    int x,w;  
  
    int y,z;  
  
    char input;  
  
    //καλούμε την initializeBoard για να αρχικοποιούμε τον πίνακα και να φτιάξουμε το γραφικό περιβάλλον  
  
    initializeBoard(array_basic_operation,&x,&y);  
  
    //παίρνοντας τις επιστρεφόμενες τιμές τις συνάρτησης initializeBoard τις κάνουμε να είναι ίδιες παλίες και καινούργιες για να  
    ξέρουμε την αρχική θέση του παίχτη μέσα στον πίνακα  
  
    W=x;  
  
    z=y;  
  
    //εδώ εκτυπώνουμε τον πίνακα, τις ζωές που έχει ο χρήστης αλλά και τους πόσους θησαυρούς έχει βρε  
  
    printBoard(array_basic_operation);  
  
    print_lifes_in_the_game(life_in_the_game);  
  
    print_tropea(found_tropea);  
  
    //δημιουργία ενός άπειρου βρόγχου
```

```
while(1){  
    do{  
        //συνάρτηση η οποία διαβάζει σε πραγματικό χρόνο τα πλήκτρα από το πληκτρολόγιο  
        input = getch();  
        //εντολή η οποία όταν αφήσει ο χρήστης το πλήκτρο σταματάει η εκτέλεση του κώδικα  
    }while(kbhit());  
    //ελέγχουμε αν το πλήκτρο που πάτησε ο χρήστης το 72 για πάνω, 80 για κάτω, 77 για αριστερά, 75 για δεξιά και 27 για  
    //τερματισμό του παιχνιδιού  
    switch (input) {  
        //κίνηση προς τα κάτω  
        case 72:  
            if (x > 0) x--;  
            break;  
        //κίνηση προς τα επάνω  
        case 80:  
            if (x < ROWS - 1) x++;
```

```
        break;  
        //κίνηση προς τα αριστερά

- case 75:
  - if (y > 0) y--;
  - break;
- //κίνηση προς τα δεξιά


- case 77:
  - if (y < COLS - 1) y++;
  - break;
- //τερματισμός παιχνιδιού με το esc -> escape


- case 27:
  - //βίαιος τερματισμός του παιχνιδιού
  - printf("try again\n");
  - printf("press enter to continue\n");
  - getchar();

```

```
        return 0;
    }

//έλεγχοι αν η καινούργια θέση που θα πάει ο χρήστης είναι προσπελάσιμη ή όχι
if(array_basic_operation[x][y]==0){

    array_basic_operation[x][y]='O';

    if(array_basic_operation[w][z]==3){

        array_basic_operation[w][z]=3;

    }else{

        array_basic_operation[w][z]=0;

    }

    printBoard(array_basic_operation);

    print_lifes_in_the_game(life_in_the_game);

    print_tropea(found_tropea);

    w=x;

    z=y;
```

```
 }else if(array_basic_operation[x][y]==2){

    array_basic_operation[x][y]=3;

    if(array_basic_operation[w][z]==3){

        array_basic_operation[w][z]=3;

    }else{

        array_basic_operation[w][z]=0;

    }

    //αυξάνουμε τους πόσους θησαυρούς έχει βρει ο χρήστης

    found_tropea[find]=1;

    //αυξάνουμε τις ζωές του χρήστη αν βρει θησαυρούς αλλά το μέγιστο είναι 5 ζωές

    if(a!=A){

        life_in_the_game[a]=1;

        a=a+1;

    }

    printBoard(array_basic_operation);
```

```
print_lifes_in_the_game(life_in_the_game);

print_tropea(found_tropea);

w=x;

z=y;

find++;

}else if (array_basic_operation[x][y]==1){

a=a-1;

//μειώνουμε τις ζωές του χρήστη αν βρει σε εμπόδιο

life_in_the_game[a]=0;

printBoard(array_basic_operation);

print_lifes_in_the_game(life_in_the_game);

print_tropea(found_tropea);

x=w;

y=z;

}else if(array_basic_operation[x][y]==3){
```

```
array_basic_operation[x][y]=3;

if(array_basic_operation[w][z]==79){

    array_basic_operation[w][z]=0;

}

printBoard(array_basic_operation);

print_lifes_in_the_game(life_in_the_game);

print_tropea(found_tropea);

w=X;

z=y;

}

//αν βρει όλους τους θησαυρούς τερματίζουμε το παιχνίδι και εμφανίζουμε χαρμόσυνο μήνυμα

if(find==C){

    printf("you find all tropea congruts\n");

    printf("press enter to conteniou\n");

    getchar();
```

```
    return 0;
}

//Αν τελειώσουν οι ζωές τερματίζουμε το παιχνίδι

if(a==0){

    printf("you lose try again\n");
    printf("press enter to conteniou\n");
    getchar();
    return 0;
}

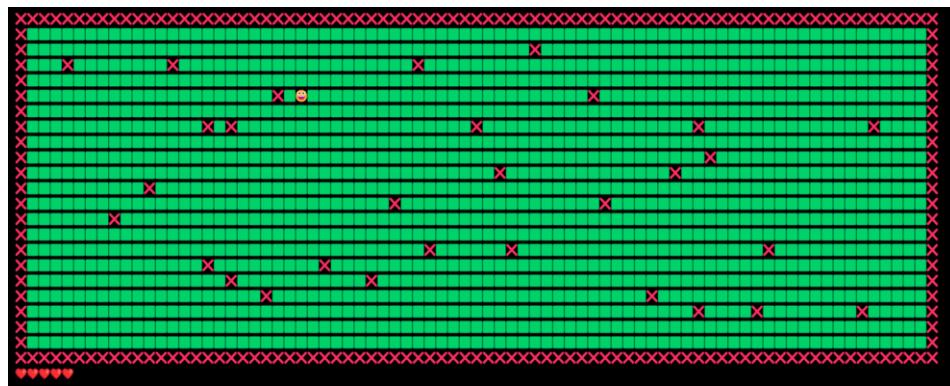
return 0;
}
```

## Εικόνες από την λειτουργεία του παιχνιδιού

Όταν ο παίχτης χάνει όλες τις ζωές του , όταν ο παίχτης βρίσκει όλους τους θησαυρούς



Όταν το παιχνίδι είναι στην αρχή



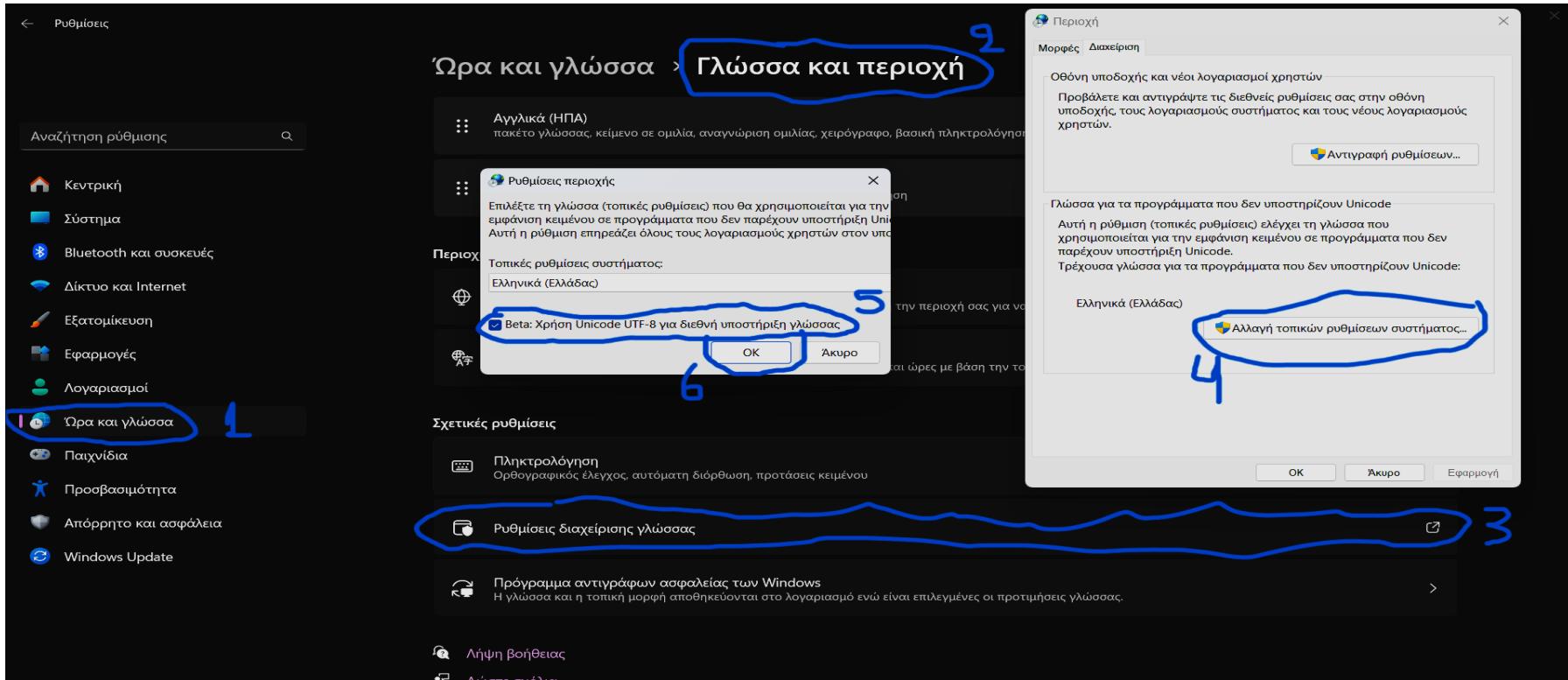
## Τρόπος λειτουργείας παιχνιδιού

Το emoji 😊 είναι ο παίκτης, το emoji ✕ είναι τα ηλεκτροφόρα εμπόδια, τα οποία κάθε φορά που τα ακουμπά ο παίκτης, χάνει μια ζωή. Το emoji ❤️ συμβολίζει τις ζωές, όπου σε κάθε αγώνα έχει 5 ζωές. Το emoji 🏆 συμβολίζει τους θησαυρούς και κάθε φορά που ο παίκτης ανακαλύπτει έναν θησαυρό, αποκτά μια ζωή. Τα emoji ■, ■ συμβολίζουν τις προσπελάσιμες θέσεις. Το ■ τις θέσεις, οι οποίες μπορεί να έχουν ή να μην έχουν θησαυρό και είναι προσπελάσιμες, ενώ το ■ τις θέσεις, οι οποίες είχαν θησαυρό και συνεχίζουν να είναι προσπελάσιμες θέσεις. Αυτό το κάνουμε γιατί θεωρούμε ότι έχουμε ένα χωράφι με ηλεκτροφόρα εμπόδια, στο οποίο πρέπει να σκάψεις για να βρεις τους θησαυρούς.

Το παιχνίδι παίζεται με τα βελάκια του πληκτρολογίου. Με το πάνω βελάκι, ο παίκτης κινείται προς τα επάνω. Με το κάτω βελάκι, ο παίκτης κινείται προς τα κάτω. Με το αριστερό βελάκι, ο παίκτης κινείται προς τα αριστερά. Με το δεξιό βελάκι, ο παίκτης κινείται προς τα δεξιά.

# Τρόπος εκτύπωσης emoji στο terminal

## Θα πρέπει να ακολουθήσετε τις παρακάτω οδηγίες:



Μόλις κάνετε όλα τα παραπάνω θα σας ζητήσει να κάνετε επανεκκίνηση τον υπολογιστή για να εφαρμοστούν οι αλλαγές (θα πρέπει να την κάνετε)

## Βιβλιογραφία - Πηγές

1. Η βοήθεια από την Ομάδα Game Development

<https://gamedev.uowm.gr/>

2. Το μάθημα δομημένος προγραμματισμός

<https://eclass.uowm.gr/courses/ICTE258/>

3. To stack overflow για την εντολή srand and rand

<https://stackoverflow.com/questions/822323/how-to-generate-a-random-int-in-c>

4. Για το παραπάνω παιχνίδι χρησιμοποιήσαμε το πρόγραμμα visual studio code και για γλώσσα προγραμματισμού την c για να κατεβάσετε το visual studio code μπορείτε να το βρείτε δωρεάν από το Microsoft store και για να ενεργοποιήσετε την γλώσσα και τον compiler θα πρέπει να ακολουθήσετε το παρακάτω YouTube video: <https://www.youtube.com/watch?v=Ubfgi4NoTPk>