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 κάνουμε ένα παιχνίδι κρυμμένου θησαυρού, στο οποίο ο παίκτης θα πρέπει να βρει 20 θησαυρούς σε έναν πίνακα 23*79 θέσεων. Υπάρχουν 30 εμπόδια μέσα στον πίνακα και στα περιθώρια του πίνακα, τα οποία είναι αδιαπέραστα, και ο τρόπος τοποθέτησής τους θα πρέπει να είναι έτσι ώστε πάντα να υπάρχει μια διαδρομή που να μπορεί ο χρήστης να ψάξει για τον θησαυρό.

Σκοπός είναι, στο τέλος, να έχουμε μάθει τα βασικά στοιχεία της C αλλά και να εμβαθύνουμε περισσότερο στην λειτουργία της εντολής rand()-> random και να αποκτήσουμε την πρώτη εμπειρία μας στο game development, όπου είναι και το θέμα που ασχολείται η ομάδα.

Η αρχική υλοποίηση του παιχνιδιού

Ανάλυση του κώδικα στην C

```
//ορίζουμε τις κύριες και βοηθητικές βιβλιοθήκες ώστε να μπορούμε να χρησιμοποιήσουμε κάποιες εντολές παρακάτω
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <time.h>

//ορίζουμε κάποιες σταθερές μεταβλητές
#define ROWS 23
#define COLS 79
#define A 20
```

```
#define B 30

//συνάρτηση η οποία τοποθετεί τον χρήστη, τα εμπόδια και τους θησαυρούς σε σωστά σημεία μέσα στον πίνακα του κύριου παιχνιδιού
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 || num_x<1 || num_y<1 ||
array_basic_operation[num_x][num_y]==1 || array_basic_operation[num_x+1][num_y]==1 ||
array_basic_operation[num_x][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-1][num_y+1]==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<A; i++){


            do{
                  num_x=rand()%(ROWS+1);
                  num_y=rand()%(COLS+1);
```

```
        }while(array_basic_operation[num_x][num_y]==1 || x==num_x || y==num_y || num_x>21 ||
num_y>77 || num_x<1 || num_y<1 || array_basic_operation[num_x][num_y]==2 ||
array_basic_operation[num_x+1][num_y]==2 || array_basic_operation[num_x][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-1][num_y+1]==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<1 || num_y<1);
	*p=num_x;
	*o=num_y;
	array_basic_operation[num_x][num_y]='O';
}

//αρχικοποιούμε τον πίνακα με όλες τις θέσεις του να έχουν τιμή 0 και ορίζουμε και τα περιθώρια του πίνακα
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;
                    }
                }
            }

    empodia(array_basic_operation,z,c);
}

//συνάρτηση εκτύπωσης του πίνακα αφού πρώτα γίνει καθαρισμός του terminal
void printBoard(int array_basic_operation[ROWS][COLS]) {
        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("O");
                }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("X");
                }
        }

    printf("\n");
  }
}
```

```
//κύρια συνάρτηση του προγράμματος
int main() {
    //ορίζουμε έναν πίνακα ακεραίων 23*79
    int array_basic_operation[ROWS][COLS];
    //ορίζουμε δύο μεταβλητές οι οποίες θα ενημερώνουν τον χρήστη για το πόσους έχει βρει  θησαυρούς αλλά και για το πόσοι του μένουν ακόμα να ανακαλύψει
    int find=0,oliko=A;
    //εντολή αρχικοποίησης της γεννήτρια παραγωγής των τυχαίων αριθμών
    srand(time(0));
    //ορίζουμε τέσσερεις ακέραιες μεταβλητές από τις οποίες το πρώτο ζεύγος αναφαίρετε στην καινούργια θέση του παίχτη μέσα στον πίνακα και το άλλο ζεύγος στην προηγούμενη θέση του παίχτη μέσα στον πίνακα
    int x,w;
    int y,z;
    //ορίζουμε μια μεταβλητή τύπου χαρακτήρα και είναι η μεταβλητή που θα αποθηκεύει το πλήκτρο που πάτησε ο χρήστης
    char input;

    //καλούμε την δεύτερη συνάρτηση που κάναμε και έχει την παρακάτω μορφή
```

```
initializeBoard(array_basic_operation,&x,&y);

//ενημερώνουμε τις παλιές θέσεις ίσες με τις καινούργιες για να ξέρουμε την αρχική θέση του παίχτη
w=x;
z=y;


//εκτυπώνουμε και καθαρίζουμε την οθόνη που βλέπει ο χρήστης με την νέα εικόνα
printBoard(array_basic_operation);
//εκτυπώνουμε πόσους θησαυρούς έχει βρει και πόσοι του μένουν ακόμα να βρει
printf("you find %d tropeo you have akoma %d\n",find,oliko-find);

//ξεκινάμε έναν άπειρο βρόγχο
while(1){

    //διαβάζουμε από το πληκτρολόγιο πιο πλήκτρο έχει πατήσει
    input = getch();
```

//συνθήκη ελέγχου για να ξέρουμε το πλήκτρο που πάτησε ο χρήστης είναι αντίστοιχο με αυτά που θέλουμε για να κινείται πάνω κάτω δεξιά και αριστερά ο παίχτης

```
switch (input) {
    //το 72 αντιστοιχεί στο πλήκτρο πάνω, το 80 αντιστοιχεί στο πλήκτρο κάτω, το 75 αντιστοιχεί στο πλήκτρο αριστερά, το 77 αντιστοιχεί στο πλήκτρο δεξιά και το 27 αντιστοιχεί στο πλήκτρο esc
    case 72:
        if (x > 0) x--;
        break;
    case 80:
        if (x < ROWS - 1) x++;
        break;
    case 75:
        if (y > 0) y--;
        break;
    case 77:
```

```
            if (y < COLS - 1) y++;
            break;
        case 27:
            printf("for close this window press enter\n");
            return 0;
}

//συνθήκες ελέγχου για την σωστή λειτουργεία του παιχνιδιού
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);
printf("congruts you find %d tropeo you have akoma %d\n",find,oliko-find);
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;
        }
        printBoard(array_basic_operation);
        w=x;
        z=y;
        printf("congruts you find %d tropeo you have akoma %d\n",find,oliko-find);
```

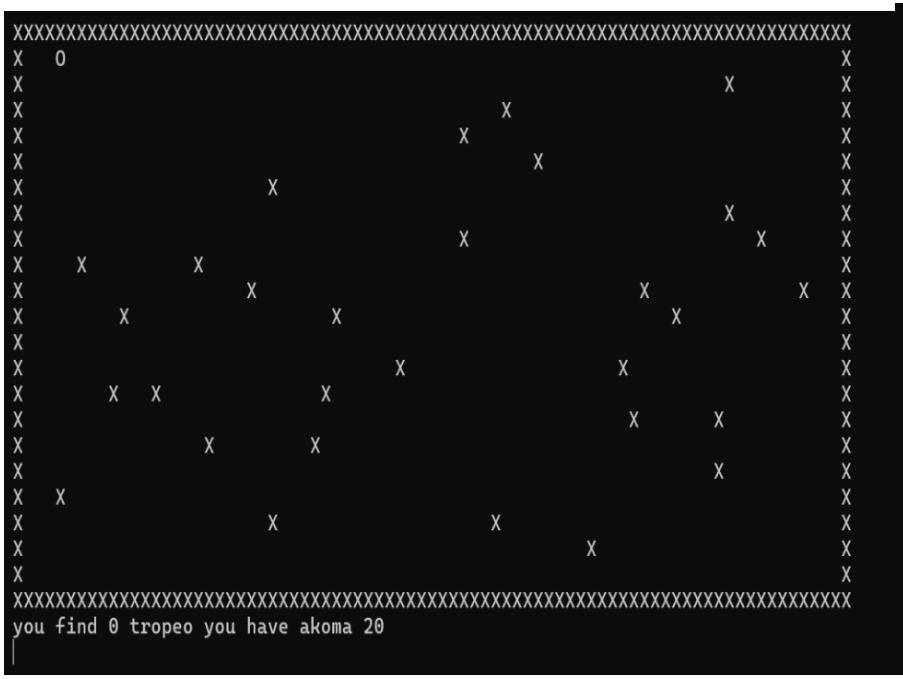
```
        find++;
}else if (array_basic_operation[x][y]==1){
        printBoard(array_basic_operation);
        printf("congruts you find %d tropeo you have akoma %d\n",find,oliko-find);
        printf("you find one empodio");
        x=w;
        y=z;
}else if(array_basic_operation[x][y]==3){
        array_basic_operation[x][y]=3;
        if(array_basic_operation[w][z]==79){
                array_basic_operation[w][z]=0;
        }
        printBoard(array_basic_operation);
        printf("congruts you find %d tropeo you have akoma %d\n",find,oliko-find);
        w=x;
```

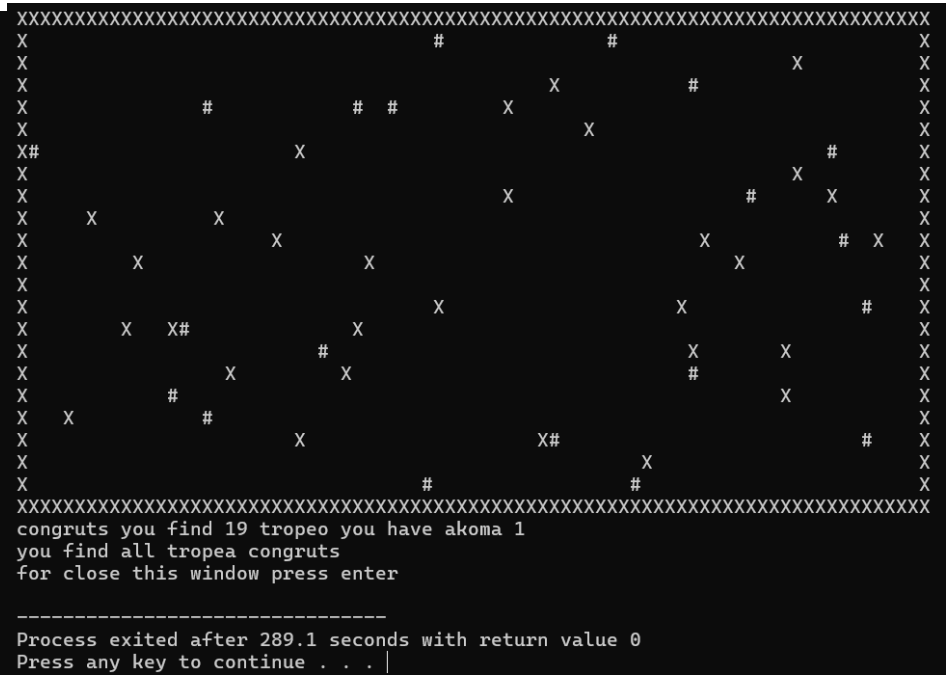
```
                z=y;
            }

            //ελέγχουμε αν ο χρήστης έχει βρει όλους τους θησαυρούς και αν ναι τερματίζουμε το παιχνίδι
            if(find==oliko){
                printf("you find all tropea congruts\n");
                printf("for close this window press enter\n");
                return 0;
            }
    }
    return 0;
}
```

Εικόνες κατά τη διάρκεια εκτέλεσης του παιχνιδιού

Εικόνα στην αρχή του παιχνιδιού και η εικόνα στο τέλος του παιχνιδιού.





Τρόπος λειτουργείας του παιχνιδιού

Το σύμβολο Ο συμβολίζει τον παίκτη μέσα στον πίνακα. Το σύμβολο Χ συμβολίζει τα εμπόδια και τα περιθώρια που έχει το παιχνίδι. Το σύμβολο # συμβολίζει ότι ο παίκτης σε εκείνη τη θέση έχει ανακαλύψει κάποιον θησαυρό. Το κενό σύμβολο, δηλαδή το space, συμβολίζει τις προσπελάσιμες θέσεις, δηλαδή τις θέσεις που μπορεί να έχουν ή να μην έχουν θησαυρό.

Ο παίκτης κινείται μέσα στον πίνακα με το πλήκτρο πάνω βέλος για να κινηθεί προς τα επάνω, με το κάτω βέλος για να κινηθεί προς τα κάτω, με το αριστερό βέλος για να κινηθεί αριστερά, και με το δεξιό βέλος για να κινηθεί δεξιά.

Ο χρήστης, αν θέλει να κάνει βίαιο τερματισμό του παιχνιδιού, μπορεί να πατήσει το πλήκτρο esc (escape) του πληκτρολογίου του, και το παιχνίδι θα τερματιστεί.

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) Το 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 dev cpp και για γλώσσα προγραμματισμού την c για εγκατάσταση του προγράμματος μπορείτε να το κατεβάσετε από το παρακάτω link: https://dev-c.soft112.com/