Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського ”

Факультет прикладної математики
Кафедра системного програмування і спеціалізованих
комп’ютерних систем

Розрахунково – графічна робота

з дисципліни

“Програмування ”

ТЕМА: “Створення ігрової програми мовою програмування С”

Група: КВ-02

Виконала: Дахал К.

Оцінка:

Київ – 2021

Завдання

Створити ігрову програму мовою програмування С.
Розробка і реалізація ігрових програм має вестися з врахуванням графічних та звукових можливостей, що надаються конкретним комп’ютером.
Програма мусить коректно розв’язувати поставлену задачу. Логічно відокремлені частини алгоритма реалізувати за допомогою окремих функцій. Також потрібно передбачити та забезпечити виконання всіх можливих розгалужень алгоритма, тобто програма повинна коректно реагувати на будь-які можливі ситуації (наприклад, виникнення помилкових ситуацій, перевірка файлів на порожність, правильність введених з клавіатури значень і т. д.). Передбачити взаємодію з користувачем (наприклад, можливість виводу правил гри, допомоги), таймер, лічильник числа ходів відповідно до поставленої в конкретному варіанті задачі.

Індивідуальний варіант

Гра-платформер. Створити персонажа та унікальну карту з блоками. Персонаж має стрибати між блоками поки не надійде до фінального блоку. Це означатиме перемогу.

Функції

check_map – перевірка чи знаходиться персонаж у рамках мапи

check_collaps – перевірка на зіткнення персонажа з блоком

map_clear – створюємо мапу

map_out – виводимо мапу

map_move_hor – рухаємо мапу горизонтально (тобто в грі рухається не персонаж, а мапа)

setcur – встановлення курсору (необхідно для коректного виводу на мапу)

position – задаємо позицію(координати)

init – ініціалізація

out – вивід

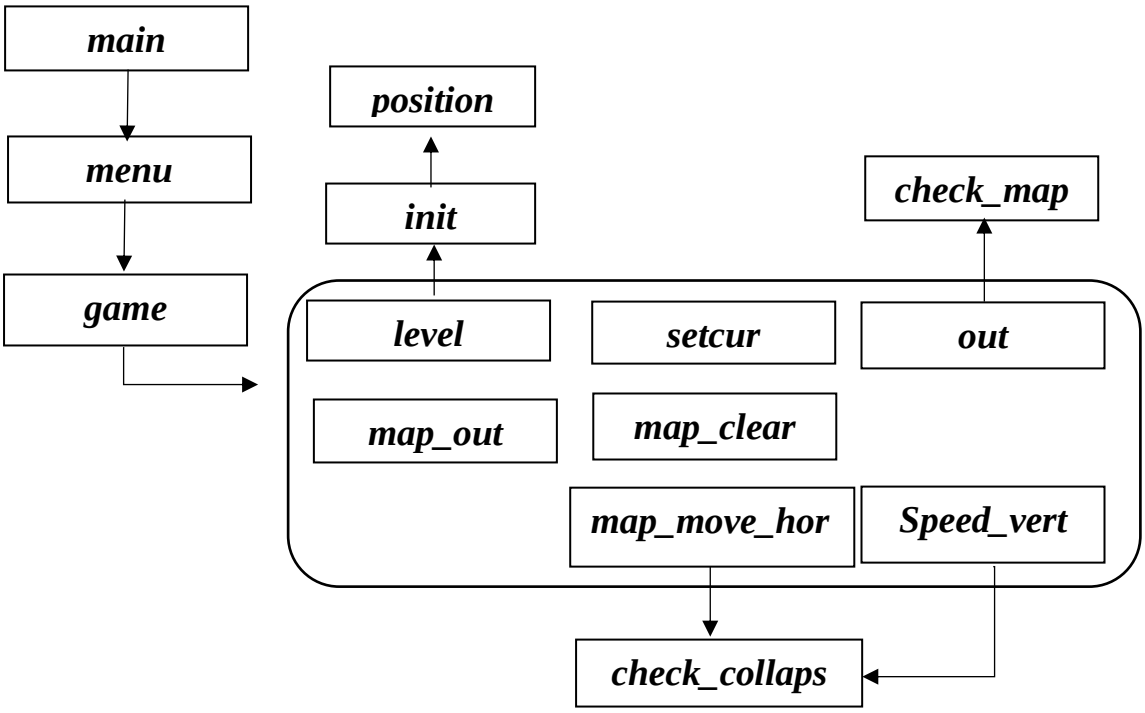
speed_vert –вертикальна швидкість

level – рівень гри, тут можна змінювати складність, міняючи форми блоків

game – об'єднуємо функції, отримуємо "чисту" гру без інтерфейсу

menu – інтерфейс

Структура



Текст програми

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <windows.h>

#define mapheight 26
#define mapwidth 120

int i, j;
int brick_lenght;
char map[mapheight][mapwidth+1];

typedef struct Sobject//оголошує"мо структуру майбутнього персонажа та блоків
{
    float x, y;//координати
    float width, height;//ширина та висота
    float speed_vert;//рух
    BOOL space_controle;//контроль стрибків
    char symbole;//чим буде намальований персонаж або блок
} Tobject;
Tobject rabbit;//наш персонаж
Tobject *brick=NULL;//блоки

BOOL check_map(int x, int y)//контроль персонажа у рамках мапи
{
    return ((x>=0)&&(x<mapwidth)&&(y>=0)&&(y<mapheight));
}

BOOL check_collaps(Tobject o1, Tobject o2)//перевірка на зіткнення з блоком
{
    return ((o1.x + o1.width)>o2.x)&&(o1.x < (o2.x + o2.width))&&
           ((o1.y + o1.height)>o2.y)&&(o1.y < (o2.y + o2.height));
}

void setcur(int x, int y)//встановлення курсору (необхідно для коректного
виводу на мапу)
{
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}

void map_clear()//створює"мо мапу
{
    for (i = 0; i<mapwidth; i++)
        map[0][i]=' ';
    map[0][mapwidth]= '\0';
    for (j = 1; j<mapheight; j++)
        sprintf(map[j],map[0]);
}
```

```
void map_out()//виводимо мапу
{
    map[mapheight-1][mapwidth-1]='\0';
    for (j = 0; j<mapheight; j++)
        printf("%s",map[j]);
}

void position(Tobject *obj, float xpos, float ypos)//задаємо
позицію(координати) персонажа
{
    (*obj).x = xpos;
    (*obj).y = ypos;
}
void init(Tobject *obj, float xpos, float ypos, float rwidth, float rheight,
char rsymbole)//ініціалізація
{
    position(obj, xpos, ypos);
    (*obj).width = rwidth;
    (*obj).height = rheight;
    (*obj).speed_vert = 0;
    (*obj).symbole = rsymbole;//символ, яким ми будемо "малювати" об'єкт
}

void out(Tobject obj)//вивід
{
    //приводимо типи та використовуємо функцію округлення, оскільки
координати мають цілий тип
    int ix = (int)round(obj.x);
    int iy = (int)round(obj.y);
    int iwidth = (int)round(obj.width);
    int iheight = (int)round(obj.height);

    for (i = ix; i<(ix + iwidth); i++)
        for (j = iy; j < (iy + iheight); j++)
            if(check_map(i, j))
                map[j][i]= obj.symbole;//вивід символів на мапу
}
void level();
void speed_vert(Tobject *obj)//вертикальна швидкість
{
    (*obj).space_controle = TRUE;
    (*obj).speed_vert += 0.05;//надаємо швидкість(по вертикалі, для стрибку)
    position(obj, (*obj).x, (*obj).y+(*obj).speed_vert);//задаємо цю
швидкість в координатах
    for (i=0; i<brick_lenght; i++)
        if (check_collaps(*obj, brick[i]))//перевірка на зіткнення з блоком
        {
            (*obj).y -=(*obj).speed_vert;//надаємо від'ємної швидкості, тобто
опускаємо карту (стрибок)
            (*obj).speed_vert = 0;//повертаємо персонажа на блок (або
коректніше: повертаємо мапу наверх)
            (*obj).space_controle = FALSE;//персонаж стоїть на блоці
            if (brick[i].symbole=='|')//якщо персонаж опиняється на блоці з
відповідних символів,
                                      //гра починається з початку
            {
```

```
                Sleep(1000);
                level();
            }
            break;
        }
}

void map_move_hor(float dx)//рухаємо мапу горизонтально (тобто в грі рухається не персонаж, а мапа)
{
    rabbit.x -= dx;//персонаж падає
    for (i=0; i<brick_lenght; i++)
        if (check_collaps(rabbit, brick[i]))
        {
            rabbit.x+=dx;//повертаємо його на поверхню
            return;//гра починається з початку
        }
    rabbit.x += dx;
    for (i=0; i<brick_lenght; i++)
        brick[i].x += dx;//саме рух мапи
}

void level()//рівень гри, тут можна змінювати складність, міняючи формат блоків та відстань між ними
{
    init(&rabbit,39, 10, 3, 3,'/');

    brick_lenght = 8;
    brick = realloc( brick, sizeof(*brick) * brick_lenght );//виділяємо память для блоків
    init(brick+0, 20, 20, 40, 5, '_');
    init(brick+1, 80, 20, 15, 10,'_');
    init(brick+2, 110, 20, 20, 5,'_');
    init(brick+3, 140, 15, 10, 10,'_');
    init(brick+4, 170, 20, 20, 5,'_');
    init(brick+5, 200, 20, 15, 15,'_');
    init(brick+6, 230, 20, 20, 5,'_');
    init(brick+7, 270, 20, 20, 5,'|');//останній блок після його гра починається з початку
}

void game()//об'єднуємо функцію, отримуємо "чисту" гру без інтерфейсу
{
    system("color 6F");//жовтий колір на задньому плані
    level();
    do
    {
        map_clear();
        //стрибаємо за допомогою клавіши "пробіл"
        if ((rabbit.space_controle == FALSE)&&(GetKeyState(VK_SPACE)<0)) rabbit.speed_vert=-1;
        if (GetKeyState('A')<0)map_move_hor(1);//рухаємося вліво за допомогою клавіши A
        if (GetKeyState('D')<0)map_move_hor(-1);//рухаєммося вліво за допомогою клавіши D
        if (rabbit.y > mapheight)//розпочинаємо з початку, якщо персонаж впав
```

```
        {
            Sleep(1000);
            level();
        }
        speed_vert(&rabbit);//щоб персонаж стояв на блоці (контроль зіткнення)
        //вивід блоків
        out(brick[0]);
        out(brick[1]);
        out(brick[2]);
        out(brick[3]);
        out(brick[4]);
        out(brick[5]);
        out(brick[6]);
        out(brick[7]);
        out(brick[8]);

        out(rabbit);//вивід персонажа
        setcur(0,0);//встановлюємо курсор
        map_out();//виводимо мапу
    }
    while (GetKeyState(VK_ESCAPE)>=0);//вихід з гри за допомогою клавіши esc
}

void menu()//інтерфейс
{
    system("color 6");
    char str [] = "\nHello, my friend\nSeems like you wanna play a little game\nWould you like to read the rules first?";
    for (int i = 0; i < strlen (str); i++)//побуквенний вивід
    {
        Sleep(30);
        printf("%c",str[i]);
    }
    int a, b;
    printf("\n1 - yes\n2 - no\n");
    scanf("%d", &a);
    switch (a)
    {
    case 1://правила
        system("color 8");
        printf("\n\nSo the rules are really easy:\n\nJump between the blocks and try not to fall in the deep,\n\n");
        printf("But if you do so, you will start the level from the very beginning.\n");
        printf("\nYou can control your character with A, D and space.\n");
        printf("\nGo to the last block and receive a victory.\n");
        printf("\nI believe in you, my friend!\n\n");
        printf("\n\npress 1 - if you are ready to go...\n");
        scanf("%d", &b);
        switch(b)
        {
        case 1:
            game();
            break;
        default :
```

```
            game();
        }
        break;

    case 2://rpa
        game();
        break;
    }
}

int main()
{
    menu();
    return 0;
}
```

Тестування

```
Hello, my friend
Seems like you wanna play a little game
Would you like to read the rules first?
1 - yes
2 - no
1


So the rules are really easy:

Jump between the blocks and try not to fall in the deep,

But if you do so, you will start the level from the very beginning.

You can control your character with A, D and space.

Go to the last block and receive a victory.

I believe in you, my friend!



press 1 - if you are ready to go...
```