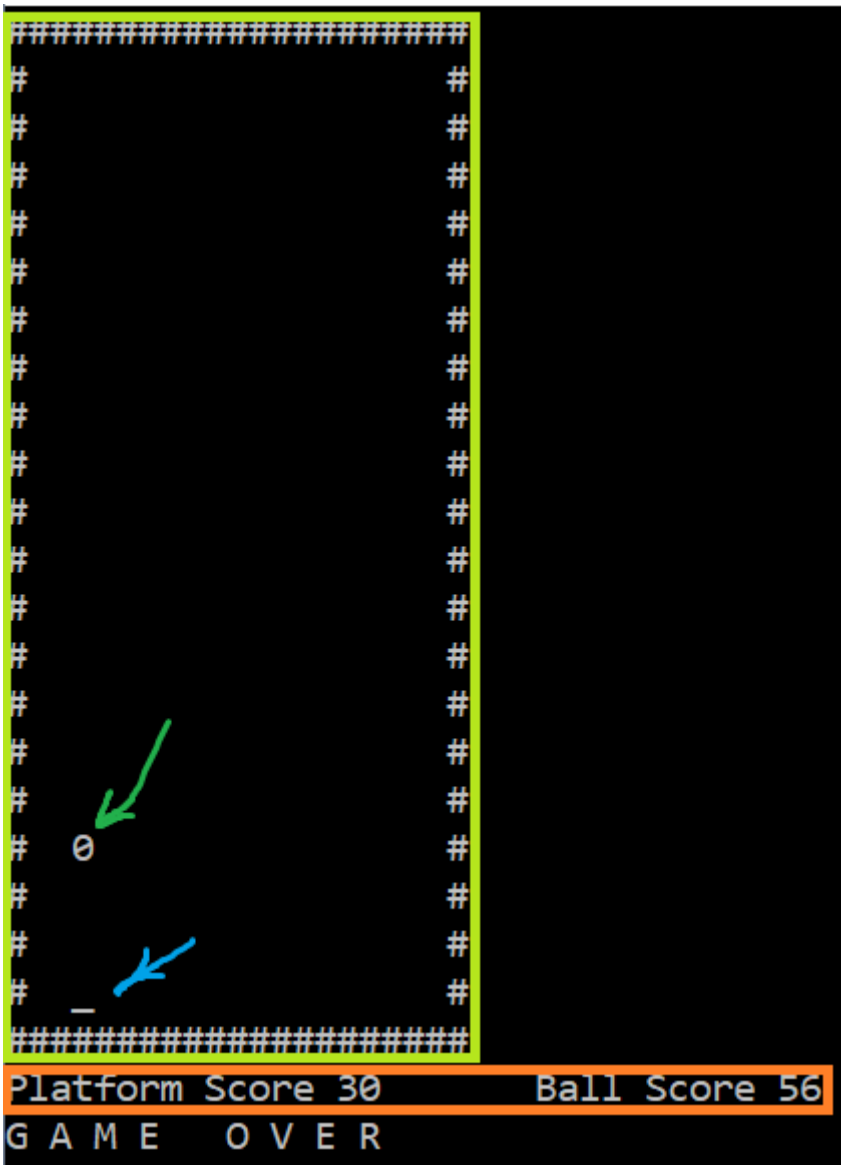
Projekt z Podstaw Programowania

Temat: Testowa wersja gry “Ping Pong”

Opis rozwiązywanego zagadnienia

Moja wersja gry ma w celu otrzymać jak najwięcej *„score”* dla strony grającej.

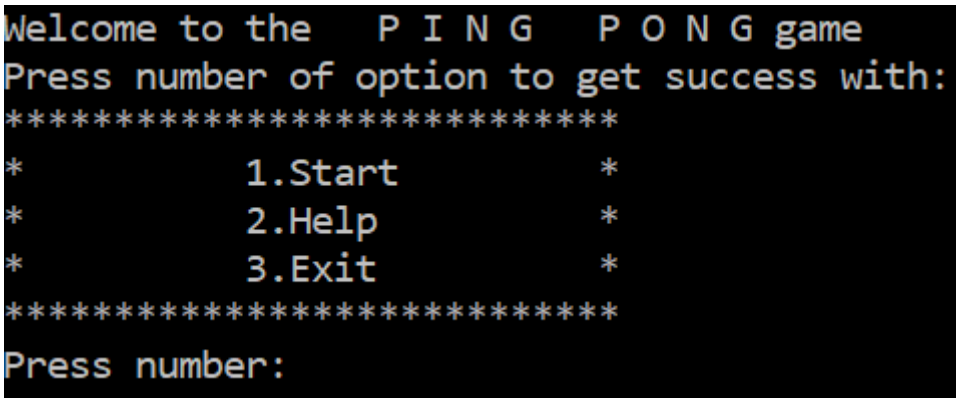
Na mapie rysowanej przez funkcje *drawing_game() znajdują się takie rzeczy:*



- Sałatowym kolorem zaznaczony jest *range* mapy (20x20)
- Zielonym kolorem zaznaczono „objekt” -> Ball
- Niebieski kolor ma „objekt” -> Platform
- Pomaranczowy prostokąt zawiera w sobie „score” dla porównowania

//Gameover na rysunku będzię wypisany w momencie kliknięncia przyciska X

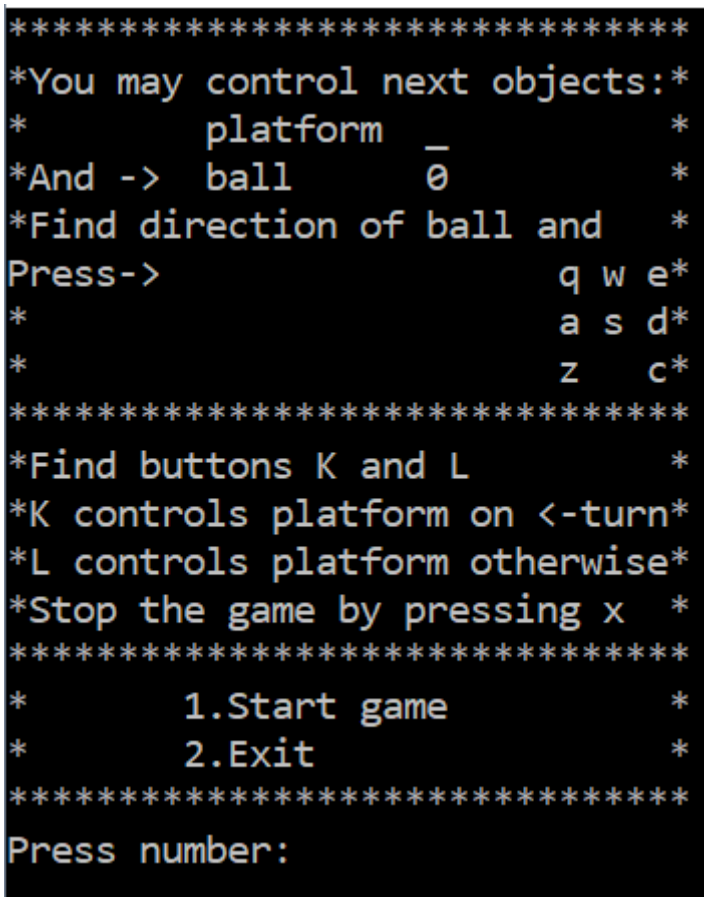
Uruchamiając program wykonyje funkcje *welcome()*



„Rekomendacja od mnie jest taka, skoro uruchomiłeś program po raz pierwszy, czytaj 'Help'"

//Podając inny numer od proponowanych -> 5 czy 10 <- Program będzie dalej potrzebował od użytkownika opcje (1) , (2) czy (3).

Klikając na punkt 2 będzię wykonana funkcja *Help()*::



Tutaj grać może zobaczyć instrukcje:

- Jakie są obiekty na mapie
- Jak kontrolować obiektami
- Jak wyłączyć program

Opis metody rozwiązania

Deklaruję zmienne ->

```
typedef enum { false, true } bool;

void restart_game ();
bool gameOver;
const int width = 20;
const int height = 20;
float x,y,platformX,platformY;
int score_platform,score_ball;
///deklaruje directy
//dir dla pilki
typedef enum directionwaza {STOP = 0, LEFT,RIGHT,UP,DOWN,UPLEFT,UPRIGHT,DOWNLEFT,DOWNRIGHT};
enum directionwaza dir;
//dir dla platformy
typedef enum dirplatform {lef_t,righ_t};
enum dirplatform dirp;
```

Robiąc funkcjami, mamy parę linii kodu w funkcje *Main()*.

Rozpatrzmy main i omówimy algorytm:

```
int main ()
{
    int one = 0;///zmienna dla swithca
    set_up();///konfiguruje wszystkie dane
    welcome(&one);///menu
    ///w zaleznosci od wpisanej liczby otrzymujemy potrzebna funkcje
    switch(one)
    {
    case 1:
        //start game funkcja
        while(!gameOver)
        {
            drawing_game();
            input();
            logic_of_game();
        }
        return 0;
        break;
    case 2:
        //HELP funcrion
        help();
        break;
    case 3:
        //EXIT function
        return 0;
        exit(0);
        break;


    }

}
```

Ważne jest że najpierw wywołana funkcja set_up()

```
///set_up funkcja konfiguruje potrzebne nam parametry
void set_up ()
{
    gameOver = false;
    dir = STOP;
///centrum mapy
    x = width / 2 - 1;
    y = height /2 - 1;
///platform x y
    platformX = width / 2 - 1;
    platformY = height - 1;
///score
    score_platform = 0;
    score_ball = 0;
}
```

Dalej w Main wywołana welcome(&one)

```
void welcome(int * o)
{
    int i_1 = 0;
    bool dop = false;
    printf("Welcome to the\t P I N G   P O N G game\n");
    printf("Press number of option to get success with:");
    printf("\n");
    printf("*******************************\n");
    printf("*          1.Start            *\n");
    printf("*          2.Help             *\n");
    printf("*          3.Exit             *\n");
    printf("*******************************\n");
    ///sprawdzam na prawidlowosc wpisanej liczby
    while(!dop)
    {
        if(i_1==1)
        {
            *o = 1;
            dop = true;
        }
        else if(i_1 == 2)
        {
            *o = 2;
            dop = true;

        }
        else if(i_1 == 3)
        {
            *o = 3;
            dop = true;
```

```
        }
        else
        {
            printf("Press number:");
            scanf("%d",&i_1);
        }
    }
}
```

Po welcome(&one) korzystajmy switch(a) , który decyduje następny krok.

//Switch jest widoczny dwa slajdy wyżej

Kluczowym kawałkiem kodu jest w wile(*!gameOver*) {

Drawing_map() ->

```
///rysuje nasza mape
void drawing_game()
{
    int i_1, i_2, j;
    system("cls");///odswieza przestrzen
    for(i_1 = 0; i_1 < width +1; i_1++)
    {

        printf("#");
    }

    printf("\n");
    for( i_2 = 0; i_2< height; i_2++)
    {
        for(j = 0; j < width; j++)
        {
            if(j == 0 || j == width - 1)
            {
                printf("#");
            }
            if(i_2 == y && j == x)
            {
                printf("0");
            }
            else if (i_2 == platformY && j == platformX )
            {

                printf("_");
            }
            else
            {
```

```
            else
            {
                bool print = false;
                if (!print)
                {

                    printf(" ");
                }

            }
        }

        printf("\n");
    }
    int i_3;
    for( i_3 = 0; i_3 < width + 1; i_3++)
    {

        printf("#");
    }

    printf("\n");
    printf("Platform Score %d",score_platform);
    printf("\t");
    printf("Ball Score %d",score_ball);
}
```

Input() ma *_kbhit() -> „if (_keybord-hit())”* , skoro jest tak -> switch który nadaje właściwości konkretnemu przycisku

```
switch(getch())
{
case 'a':
    dir = LEFT;

    break;
case 'd':
    dir = RIGHT;
    break;
case 'w':
    dir = UP;
    break;
case 's':
    dir = DOWN;
    break;
case 'q':
    dir = UPLEFT;
    break;
case 'e':
    dir = UPRIGHT;
    break;
case 'z':
    dir = DOWNLEFT;
    break;
case 'c':
    dir = DOWNRIGHT;
    break;
case 'k':
    dirp = lef_t;
    break;
case 'l':
    dirp = righ_t;
    break;
case 'x':
    printf("\nG A M E   O V E R");
    gameOver = true;
```

Logic_of_game jest długa funkcja .

„Jeśli main to jest Głowa naszej gry, to funkcja l_o_g -> świadomość, myślenie”

- Kawałek switch(dir) nadaje właściwości zmiennym x oraz y dla obiektu Ball

```
switch(dir)
{
case LEFT:
    x--;
    break;
case RIGHT:
    x++;
```

- Kawałek switch(dirp) nadaje właściwości zmiennej platformX

```
switch(dirp)
{
case lef_t:
    platformX -=1;
    break;
case righ_t:
    platformX +=1;
    break;
}
```

- Nadajemy kierunki dla osi X

```
if (x >= width - 1)//sprawdzanie jezeli x juz jest na linii czy poza granicy
{
    if (dir == DOWNRIGHT)//jezeli direction Ball(a) jest diagonal w prawy dół
    {
        dir = DOWNLEFT;//dir staje diagonala odrotnej dolnej strony
    }
    else if (dir == UPRIGHT)//tak samo sprawdzenie na kierunek w diagonal gory
    {
        dir = UPLEFT;//zmieniamy kierunek na lewa strone ,ale dalej w góre
    }
    else//zostaje tylko kierunek  pianowo ,więć do tej pory  kierunek był right
    {
        dir = LEFT;//tutaj kierunek zmieniamy na LEFT
    }
}
else if (x < 0)//sprawdzanie jezeli x juz jest na linii czy poza granica
{
    if (dir == DOWNLEFT)// i tak samo dla lewej granicy
    {
        dir = DOWNRIGHT;
    }
    else if(dir == UPLEFT)
    {
        dir = UPRIGHT;
    }
    else
    {
        dir = RIGHT;
    }
}
```

- Nadajemy kierunek dla osi Y

```
if (y >= height) // jezeli wyzej czy rowno dolnej granicy
{
    //gameOver = true;//szybki sposob zakonczyc gre:)
    score_ball +=2;// + score dla ball(a)
    if(dir == DOWNRIGHT)//jezeli kierunek byl w prawy dół
    {
        dir = UPRIGHT;//zmieniamy na prawa góre
    }
    else if(dir == DOWNLEFT) // jeśli kierunek był w lewy dół
    {
        dir = UPLEFT;//zmieniamy na w góre na lewo
    }
    else//zostaje nam kierunek poziomowy w dół
    {
        dir = UP;//zmienia sie na góre
    }
}
else if (y < 0)
{
    if(dir == UPRIGHT)
    {
        dir = DOWNRIGHT;
    }
    else if(dir == UPLEFT)
    {
        dir = DOWNLEFT;
    }
    else
    {
        dir = DOWN;
    }
```

- Nadajemy kierunki dla platformy

```
if(platformX >= width - 1)//jezeli X platrofrmy wyjechal lub spotkał  prawa granice
{
    if(dirp == righ_t)// jesli keirunek byl w prawo
    {
        dirp = lef_t;//zmieniamy w lewo
    }
}
else if (platformX <= 0)//jesli X platfromy wyjechal lub spotkal lewa granice
{
    if(dirp == lef_t)//czy kierunek byl w lewo
    {
        dirp = righ_t;//zmieniamy w prawo
    }
}
```

- Spotkanie dwóch obiektów

```
if (x == platformX && y == platformY)//jesli BALL wpadnie na PLATFORM//czyli platforma odbila BALL
{
    score_platform += 10;//score dla gracza za platforme +10 bo to jest ciężko
    if(dir == DOWNRIGHT)//jesli kierunek  objektu BALL(a)
    {
        dir = UPRIGHT;//zmieniamy
    }
    else if(dir == DOWNLEFT)//jesli kierunek był
    {
        dir = UPLEFT;//staje tako
    }
    else//zostaje poziom
    {
        dir = UP;//zmieniamy w góre
    }
}
```

}//koniec funkcji l_o_g

Kod funkcji Help() ->

- Rysuję okno

```
void help()
{
    int i = 0;
    bool dop = false;
    system("cls");
    printf("**********************************\n");
    printf("*You may control next objects:*\n");
    printf("*        platform  _             *\n");
    printf("*And ->  ball      0             *\n");
    printf("*Find direction of ball and      *\n");
    printf("*Press->                   q w e*\n");
    printf("*                          a s d*\n");
    printf("*                          z   c*\n");
    printf("**********************************\n");
    printf("*Find buttons K and L            *\n");
    printf("*K controls platform on <-turn*\n");
    printf("*L controls platform otherwise*\n");
    printf("*Stop the game by pressing x     *\n");
    printf("**********************************\n");
    printf("*       1.Start game              *\n");
    printf("*       2.Exit                    *\n");
    printf("**********************************\n");
```

- Sprawdzamy na prawidłowość wpisanej liczby

```
while(!dop)
{
    if(i == 1)
    {
        dop = true;
        while(!gameOver)
        {
            drawing_game();
            input();
            logic_of_game();
        }
    }
    else if(i == 2)
    {
        return 0 ;
        exit(0);
    }
    else
    {
        printf("Press number:");
        scanf("%d",&i);
    }
}
```

Informacje na temat środowiska uruchomieniowego

System Operacyjny

- Windows 10

Oprogramowanie

- Code::Bloks 16.01

Compilator

- C99

Kontakt

W razie jakichkolwiek pytań i wątpliwości pisać na *vibeyesworkspace@gmail.com*