game

Autor: Tamara Davidović
Univerzitet Singidunum
///datum
Novi Sad

Osnovni podaci

- **Programski jezik: C++**
- **Razvojno okruženje: Code::Blocks**
- **Biblioteke: SDL2, SDL_TTF, SDL_image**

Cilj igre

- Sakupiti što više crvenih krugova, sačuvati ih od protivnika.
- Svaki sakupljeni krug uvećava rezultat za 1.

- Ukolko automatsi igrač stigne do crvenog kruga pre igrača rezultat se smanju
- Ukoliko automatski igrač stine igrača korisnika poeni se vraćaju na 0



Crveni krug – KLASA BOMBONA

bombona.h

```
20  class Bombona:public Sprite{
21  public:
22      Bombona(SpriteSheet *sheet, int width = 64, int height = 64);
23      void draw(SDL_Renderer * renderer);
24      void move();
25      void move (int dx, int dy);
26
27
28      int randomBroj();
29
30  };
```

bombona.cpp

```
46
47  int Bombona::randomBroj(){
48          srand(time(NULL));
49          int br = rand() % 40;
50          return br;
51
52
53  }
54
```

IGRAČ KORISNIK

igrac.h

```
20
21  class Igrac:public Sprite{
22  public:
23      Igrac(SpriteSheet *sheet, int width = 64, int height = 64);
24      void draw(SDL_Renderer * renderer);
25      void move();
26      void move (int dx, int dy);
27
28      void load();
29      void save();
30  };
```

igrac.cpp

```
65
66  void Igrac::save(){
67      ofstream myfile;
68      myfile.open("save/file.txt");
69      myfile << spriteRect->x << "\n" << spriteRect->y << "\n";
70  }
```

```
71
72  void Igrac::load(){
73      ifstream myfile;
74      myfile.open("save/file.txt");
75
76      if(!myfile.is_open()){
77          exit(EXIT_FAILURE);
78      }
79      myfile >> spriteRect->x;
80      myfile >> spriteRect->y;
81  }
```

IGRAČ RAČUNAR

igracracunar.h

```
20
21  class IgracRacunar:public Sprite{
22  public:
23      IgracRacunar(SpriteSheet *sheet, int width = 64, int height = 64);
24      void draw(SDL_Renderer * renderer);
25      void move();
26      void move (int dx, int dy);
27
28      //vraca jedan od random brojeva 1 2 4 8
29      int randomBroj();
30      void changeState();
31
32      int st;
33      int ubrzanje = 5;
34      Bombona *b;
35
36  };
```

IGRAČ RAČUNAR

igracracunar.cpp

```
31
32  void IgracRacunar::move(int dx, int dy) {
33      spriteRect->x += ubrzanje*dx;
34      spriteRect->y += ubrzanje*dy;
35
36      ///ograniceno kretanje
37      if (spriteRect->x < 0){
38          spriteRect->x = 0;
39          changeState();
40      }
41      else if (spriteRect->x > 640 - spriteRect->w){
42          spriteRect->x = 640-spriteRect->w;                  //640 sirina prozora
43          changeState();
44      }
45      if ( spriteRect->y < 0){
46          spriteRect->y =0;
47          changeState();
48      }
49      else if (spriteRect->y > 480 - spriteRect->h){          //480 visina prozora
50          spriteRect->y = 480 - spriteRect->h;
51          changeState();
52      }
53
54
55  }
```

IGRAČ RAČUNAR

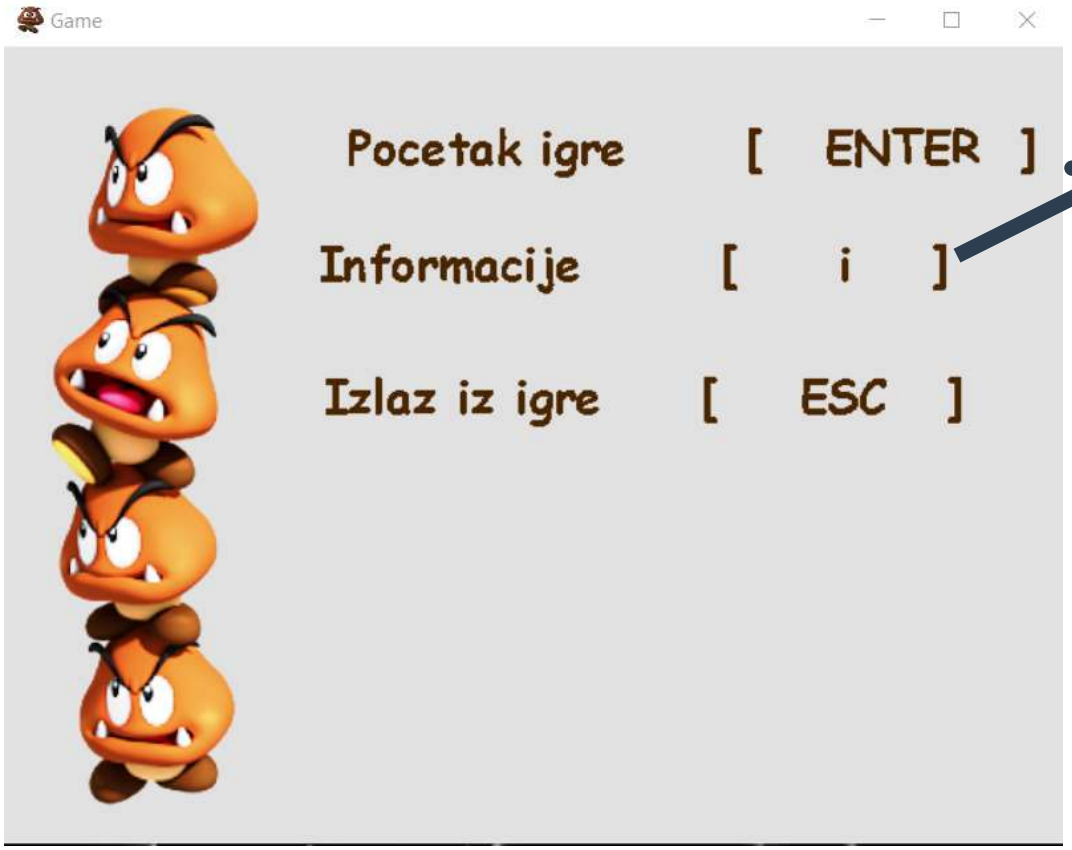
igracracunar.cpp

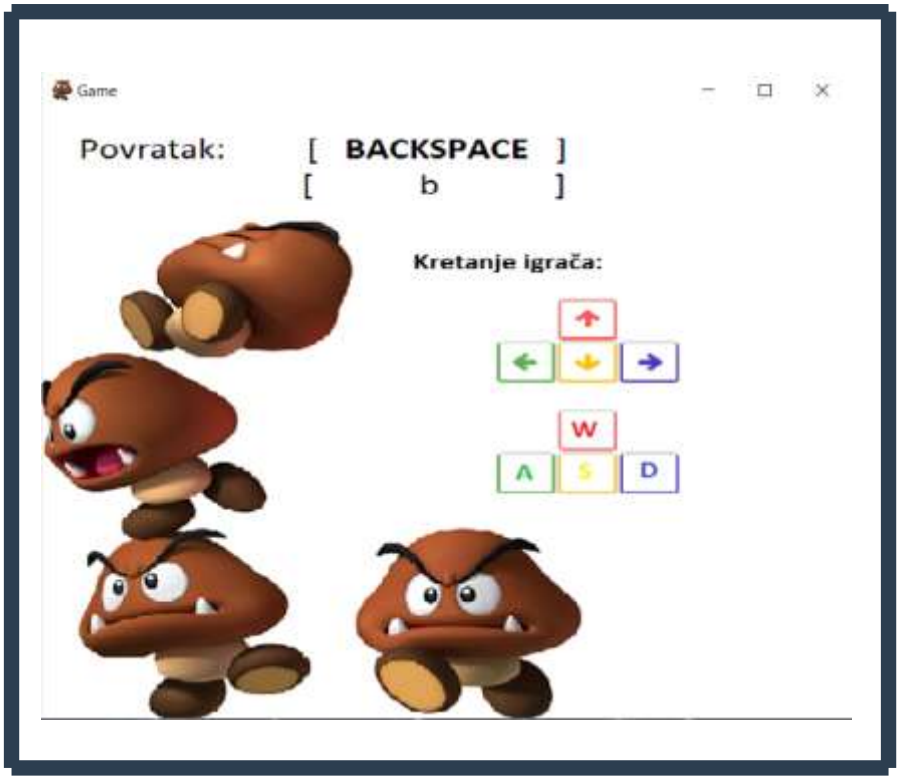
```
73
74  int IgracRacunar::randomBroj(){
75          srand(time(NULL));
76          int br = rand() % 4;
77          switch(br){
78          case 0:
79              return 1;
80              break;
81          case 1:
82              return 2;
83              break;
84          case 2:
85              return 4;
86              break;
87          case 3:
88              return 8;
89              break;
90          default:
91              return 0;
92              cout << "Doslo je do greske!!";
93          }
94
95  }
```

```
96
97  void IgracRacunar::changeState(){
98      do {
99          st = randomBroj();
100         break;
101     }while(st == state);
102         setState(st);
103
104 }
105
```

** ako ne stavim [break]
igrač ne udara u ivicu, ali zbog while petlje, igrica će u nekom trenutku početi da baguje...

Početni ekran





Početni ekran

```
36  class PocetniEkran {
37
38  public:
39
40
41      int frameCap = 60;
42      PocetniEkran(string title);
43      void addDrawable(Drawable* drawable);
44
45      void run();
46
47      virtual ~PocetniEkran();
48
49      void listen(SDL_KeyboardEvent &event);
50
51      void ekranIgrice();
52      void tasteri(SDL_KeyboardEvent *key);
53      void zatvoriPocetniEkran();
54      void pocetakIgreTekst(TTF_Font *font, SDL_Renderer *renderer);
55      void izlazakIzIgreTekst(TTF_Font *font, SDL_Renderer *renderer);
56      void informacijeTekst(TTF_Font *font, SDL_Renderer *renderer);
57      void prikaziInformacije(TTF_Font *font, SDL_Renderer *renderer);
58
59      SDL_Window* window = NULL;
60      SDL_Surface* background_surface = NULL;
61      SDL_Texture* background_texture = NULL;
62      SDL_Renderer* renderer = NULL;
63
64      bool info = false;
65
66  };
67
```

Vrste igrača

Igrač kojim upravlja korisnik:

Igrači kojim upravlja računar:

Nivo 1

- Postoje tri automatska igrača, različitih brzina kretanja.
- Da bi se prešlo na sledeći nivo potrebno je sakutpiti 5 poena.

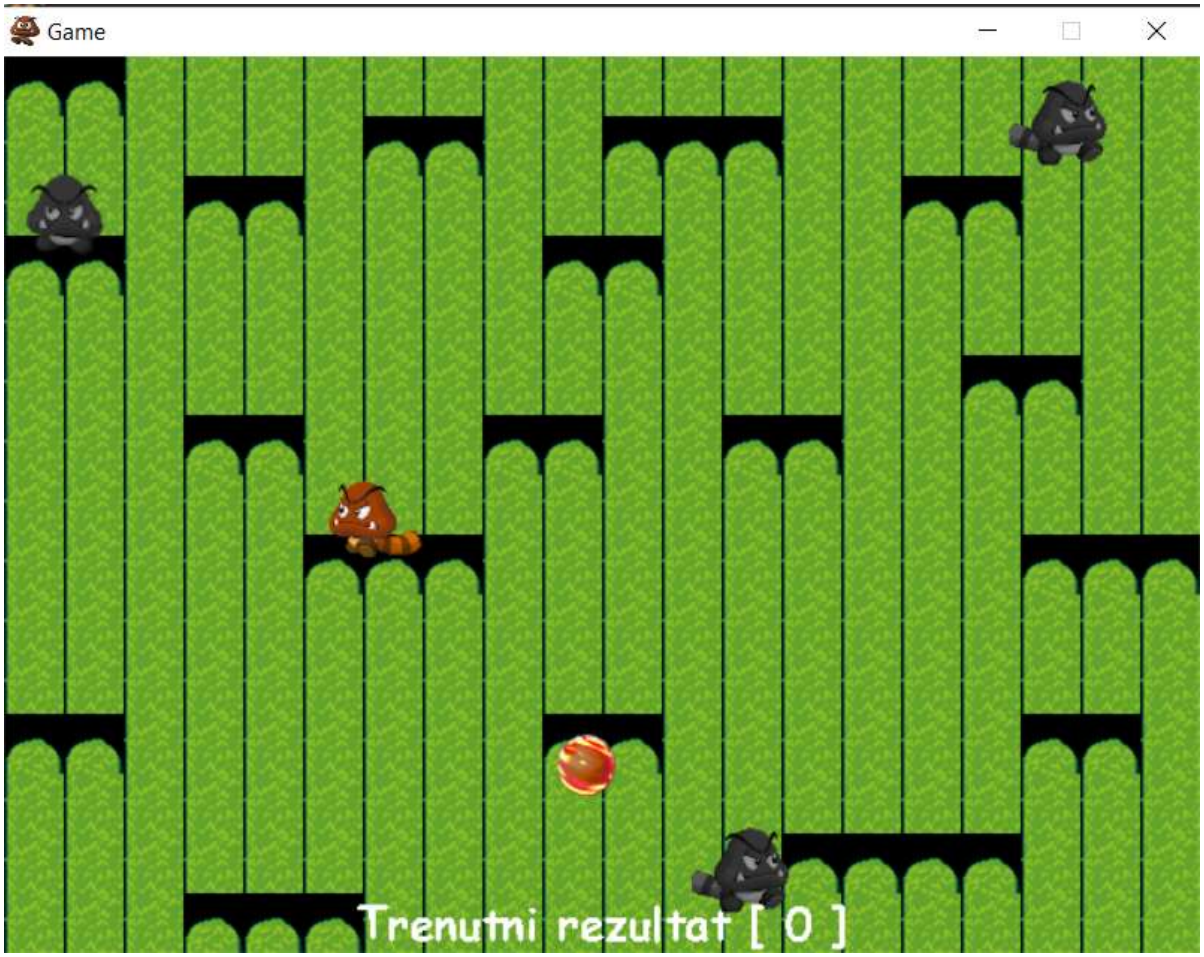
Ispis rezultata

Nivo

```
71
72      void run();
73      void nexLevel();
74      void nexLevel2();
75
```

```
77
78      void drawScore(int score, SDL_Renderer* renderer);
79      void ispitivanjeKolizija();
80
```

Nivo 2



- Dolazi do promene pozadine.
- Postoje tri automatska igrača, kre
- Da bi se prešlo na sledeći nivo po

Nivo 3

- Dolazi do promene pozadine.
- Postoje tri automatska igrača, kreću
- Ne postoji ograničenje rezultata.

Funkcija random:

```
9   #include <stdlib.h>
10  #include <time.h>
11
```

```
74  int IgracRacunar::randomBroj(){
75          srand(time(NULL));
76          int br = rand() % 4;
77          switch(br){
78          case 0:
79              return 1;
80              break;
81          case 1:
82              return 2;
83              break;
84          case 2:
85              return 4;
86              break;
87          case 3:
88              return 8;
89              break;
90          default:
91              return 0;
92              cout << "Doslo je do greske!!";
93          }
94
95  }
96
```

```
46
47  int Bombona::randomBroj(){
48          srand(time(NULL));
49          int br = rand() % 40;
50          return br;
51
52
53  }
54
```

```
96
97  void IgracRacunar::changeState(){
98      do {
99          st = randomBroj();
100         break;
101     }while(st == state);
102         setState(st);
103
104 }
105
106
```

```
52
53  int  Group::kolizijaBombona(Igrac* igracKorisnik, Bombona* b){
54      int xKorisnik = igracKorisnik->spriteRect->x;
55      int xB = b->spriteRect->x;
56      int yKorisnik = igracKorisnik->spriteRect->y;
57      int yB = b->spriteRect->y;
58
59      int x = xKorisnik - xB;
60      int y = yKorisnik - yB;
61      int distanca = sqrt(x*x + y*y);
62
63      if ( distanca < 16+16 ){
64          cout<< "BOMBOOOOOOOOOOOOOOOOOONA!\n";
65          scor ++;
66          cout << scor;
67          //case fuju za 5-6 slucajeva, i ond random uzmem da izaberem gde da stane
68
69          b->spriteRect->x = b->randomBroj()*10;
70          b->spriteRect->y = b->randomBroj() *8;
71
72
73          }
74      return scor;
75  }
```

Funkcije za koliziju:

```
17
18
19  /** \brief Klasa koja je zaduCena za detekciju kolizije, i dektekciju da li je igrac u okviru prozora.
20   */
21  class Group {
22  public:
23      Group();
24      int scor = 0;
25      void kolizija(Igrac *igracKorisnik, IgracRacunar* igracRacunar);
26      int kolizijaBombona(Igrac *igracKorisnik, Bombona* b);
27      void  kolizijaBombonaAutomatskiIgrac(IgracRacunar* ir, Bombona* b);
28      void kolizijaAutomatskihIgraca(IgracRacunar *igracRacunar1, IgracRacunar *igracRacunar2);
29
30
31  };
32
```

Kako i gde pozivan funkcije:

```
274
275 void Engine :: ispitivanjeKolizija(){
276     g->kolizijaAutomatskihIgraca(igracRacunar, igracRacunar2);
277     g->kolizijaAutomatskihIgraca(igracRacunar, igracRacunar3);
278     g->kolizijaAutomatskihIgraca(igracRacunar2, igracRacunar3);
279     g->kolizija(igrac, igracRacunar);
280     g->kolizija(igrac, igracRacunar2);
281     g->kolizija(igrac, igracRacunar3);
282     g->kolizijaBombonaAutomatskiIgrac(igracRacunar, b);
283     g->kolizijaBombonaAutomatskiIgrac(igracRacunar2, b);
284     g->kolizijaBombonaAutomatskiIgrac(igracRacunar3, b);
285
286
287 }
```

Funkcije za koliziju:

```
11
12  void Group::kolizija(Igrac* igracKorisnik, IgracRacunar* igracRacunar){
13      int xKorisnik = igracKorisnik->spriteRect->x;
14      int xRacunar = igracRacunar->spriteRect->x;
15      int yKorisnik = igracKorisnik->spriteRect->y;
16      int yRacunar = igracRacunar->spriteRect->y;
17
18      int x = xKorisnik - xRacunar;
19      int y = yKorisnik - yRacunar;
20      int distanca = sqrt(x*x + y*y);
21
22      if ( distanca < 16+16 ){
23          cout<< "Uhvacen!\n";
24          scor =0;
25          cout << scor;
26          //case fuju za 5-6 slucajeva, i ond random uzmem da izaberem gde da stane
27          int slucaj = igracRacunar->randomBroj();
28          switch(slucaj) {
29              case 1:
30                  igracRacunar->spriteRect->x = 2;
31                  igracRacunar->spriteRect->y = 4;
32                  break;
33              case 2:
34                  igracRacunar->spriteRect->x = 200;
35                  igracRacunar->spriteRect->y = 480;
36                  break;
37              case 4:
38                  igracRacunar->spriteRect->x =400;
39                  igracRacunar->spriteRect->y = 400;
40                  break;
41              case 8:
42                  igracRacunar->spriteRect->x = 170;
43                  igracRacunar->spriteRect->y = 15;
44                  break;
45              default:
46                  igracRacunar->spriteRect->x = 23;
47                  igracRacunar->spriteRect->y = 232;
48          }
49      }
50  }
51
```

Funkcije za koliziju:

```
51
52  int  Group::kolizijaBombona(Igrac* igracKorisnik, Bombona* b){
53      int xKorisnik = igracKorisnik->spriteRect->x;
54      int xB = b->spriteRect->x;
55      int yKorisnik = igracKorisnik->spriteRect->y;
56      int yB = b->spriteRect->y;
57
58      int x = xKorisnik - xB;
59      int y = yKorisnik - yB;
60      int distanca = sqrt(x*x + y*y);
61
62      if ( distanca < 16+16 ){
63          cout<< "BOMBOOOOOOOOOOOOOOOOONA!\n";
64          scor ++;
65          cout << scor;
66          //case fuju za 5-6 slucajeva, i ond random uzmem da izaberem gde da stane
67
68          b->spriteRect->x = b->randomBroj()*10;
69          b->spriteRect->y = b->randomBroj() *8;
70
71
72          }
73      return scor;
74  }
75
```

Funkcije za koliziju:

```
75
76  void  Group::kolizijaBombonaAutomatskiIgrac(IgracRacunar* igracRacunar, Bombona* b){
77
78      int xIR = igracRacunar->spriteRect->x;
79      int xB = b->spriteRect->x;
80      int yIR = igracRacunar->spriteRect->y;
81      int yB = b->spriteRect->y;
82
83      int x = xIR - xB;
84      int y = yIR - yB;
85      int distanca = sqrt(x*x + y*y);
86
87      if ( distanca < 16+16 ){
88          cout<< "ukrao ti je bombonu ----------------------------!\n";
89          scor --;
90          cout << scor;
91          //case fuju za 5-6 slucajeva, i ond random uzmem da izaberem gde da stane
92
93          b->spriteRect->x = b->randomBroj()*8;
94          b->spriteRect->y = b->randomBroj() *10;
95          igracRacunar->spriteRect->x = 170;
96          igracRacunar->spriteRect->y = 15;
97
98
99          }
100 }
101
```

Funkcije za koliziju:

```
101
102 void Group::kolisijaAutomatskihIgraca(IgracRacunar* igracRacunar1, IgracRacunar* igracRacunar2){
103     int xRacunar1 = igracRacunar1->spriteRect->x;
104     int xRacunar2 = igracRacunar2->spriteRect->x;
105     int yRacunar1 = igracRacunar1->spriteRect->y;
106     int yRacunar2 = igracRacunar2->spriteRect->y;
107
108     int x = xRacunar1 - xRacunar2;
109     int y = yRacunar1 - yRacunar2;
110     int distanca = sqrt(x*x + y*y);
111
112     if ( distanca < 16+16 ){
113         cout<< "Udvaceni!\n";
114         scor =0;
115         cout << scor;
116         //case fuju za 5-6 slucajeva, i ond random uzmem da izaberem gde da stane
117         int slucaj = igracRacunar1->randomBroj();
118         switch(slucaj) {
119             case 1:
120                 igracRacunar1->spriteRect->x = 2;
121                 igracRacunar1->spriteRect->y = 4;
122                 igracRacunar2->spriteRect->x = 200;
123                 igracRacunar2->spriteRect->y = 400;
124                 break;
125             case 2:
126                 igracRacunar1->spriteRect->x = 200;
127                 igracRacunar1->spriteRect->y = 480;
128                 igracRacunar2->spriteRect->x = 20;
129                 igracRacunar2->spriteRect->y = 40;
130                 break;
131             case 4:
132                 igracRacunar1->spriteRect->x =400;
133                 igracRacunar1->spriteRect->y = 400;
134                 igracRacunar2->spriteRect->x = 200;
135                 igracRacunar2->spriteRect->y = 352;
136                 break;
137             case 8:
138                 igracRacunar1->spriteRect->x = 170;
139                 igracRacunar1->spriteRect->y = 15;
140                 igracRacunar2->spriteRect->x = 10;
141                 igracRacunar2->spriteRect->y = 52;
142                 break;
143             default:
144                 igracRacunar1->spriteRect->x = 23;
145                 igracRacunar1->spriteRect->y = 232;
146                 igracRacunar2->spriteRect->x = 2;
147                 igracRacunar2->spriteRect->y = 32;
148         }
149
150
151     }
152 }
153
```

Hvala na pažnji