LAPORAN FINAL PROJECT

KONSEP PEMROGRAMAN

GAME SPACY

DISUSUN OLEH :

BIMO ADRIAN S	M0517011
DWIKI RAYYANA A	M0517012
YAUMI AZIIZAH Z	M0517052

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS SEBELAS MARET SURAKARTA

SENIN, 8 JANUARI 2018

ANALISIS *SOURCECODE*

```
1   #include <iostream>
2   #include <allegro5/allegro.h>
3   #include <allegro5/allegro_primitives.h>
4   #include <allegro5/allegro_font.h>
5   #include <allegro5/allegro_ttf.h>
6   #include <allegro5/allegro_image.h>
7   #include <allegro5/allegro_audio.h>
8   #include <allegro5/allegro_acodec.h>
9   #include "objects.h"
```

Terdapat berbagai file header yang telah didaftarkan, file header allegro yang menyimpan fungsi-fungsi untuk membuat graphic. Untuk file header “objects.h” merupakan file sc tersendiri untuk dipanggil dalam program ini.

```
12  // globals
13
14  const int WIDTH = 800;
15  const int HEIGHT = 400;
16  const int NUM_BULLETS = 5;
17  const int NUM_COMETS = 10;
18  const int NUM_EXPLOSIONS = 5;
19
20  const char FONT_DAUNPENH[] = "daunpenh.ttf";
21
22  enum KEYS { UP, DOWN, LEFT, RIGHT, SPACE };
23  bool keys[5] = { false, false, false, false, false };
```

Berfungsi untuk mendeklarasikan variabel global, seperti WIDTH atau lebar layar dideklarasikan konstan bernilai 800, lalu terdapat karakter font yang dideklarasikan dari file daunpenh.ttf

```
25  // prototypes
26  void InitBg(Background &bg, int x, int y, int width, int height, ALLEGRO_BITMAP *image);
27  void DrawBg(Background &bg);
28
29  void InitShip(SpaceShip &ship, ALLEGRO_BITMAP *image);
30  void ResetShipAnimation(SpaceShip &ship, int position);
31  void DrawShip(SpaceShip &ship);
32  void MoveShipUp(SpaceShip &ship);
33  void MoveShipDown(SpaceShip &ship);
34  void MoveShipLeft(SpaceShip &ship);
35  void MoveShipRight(SpaceShip &ship);
36
37  void InitBullet(Bullet bullet[], int size);
38  void DrawBullet(Bullet bullet[], int size);
39  void FireBullet(Bullet bullet[], int size, SpaceShip &ship);
40  void UpdateBullet(Bullet bullet[], int size);
41  void CollideBullet(Bullet bullet[], int bSize, Comet comets[], int cSize, SpaceShip &ship, Explosion explosions[], int eSize);
42
43  void InitComet(Comet comets[], int size, ALLEGRO_BITMAP *image);
44  void DrawComet(Comet comets[], int size);
45  void StartComet(Comet comets[], int size);
46  void UpdateComet(Comet comets[], int size);
47  void CollideComet(Comet comets[], int cSize, SpaceShip &ship, Explosion explosions[], int eSize);
48
49  void InitExplosions(Explosion explosions[], int size, ALLEGRO_BITMAP *image);
50  void DrawExplosions(Explosion explosions[], int size);
51  void StartExplosions(Explosion explosions[], int size, int x, int y);
52  void UpdateExplosions(Explosion explosions[], int size);
```

Merupakan sourcecode prototype fungsi digunakan saat fungsi main memanggil prototype tersebut, akan tetapi prototype fungsi harus sudah dibuat di luar fungsi main.

```
54  int main(int argc, char **argv) {
55
56      // primitive variable
57      bool done = false;
58      bool redraw = true;
59      const int FPS = 60;
60      bool isGameOver = false;
61      int menu =1;
62      Background Menu;
63
64      //Main Menu
65      int state = 1;
66
67      // object variables
68      SpaceShip ship;
69      Bullet bullets[NUM_BULLETS];
70      Comet comets[NUM_COMETS];
71      Explosion explosions[NUM_EXPLOSIONS];
```

Merupakan fungsi utama program yang akan dibuat. Mendeklarasikan variabel terlebih dahulu sesuai yang ada pada library allegro atau primitive variable, mendeklarasikan variabel yang nantinya akan menjadi objek seperti ship, bullets, comets, dan explosions yang telah dibuat sourcecodenya sendiri pada library “object.h”.

```
73      // allegro variables
74      ALLEGRO_DISPLAY *display = NULL;
75      ALLEGRO_EVENT_QUEUE *event_queue = NULL;
76      ALLEGRO_TIMER *timer = NULL;
77      ALLEGRO_FONT *font25 = NULL;
78      ALLEGRO_BITMAP *shipImage;
79      ALLEGRO_BITMAP *cometImage;
80      ALLEGRO_BITMAP *expImage;
81      ALLEGRO_BITMAP *mnImage;
82      ALLEGRO_SAMPLE *sample = NULL;
83      ALLEGRO_SAMPLE *sample2 = NULL;
84      ALLEGRO_SAMPLE *sample3 = NULL;
85      ALLEGRO_SAMPLE_INSTANCE *instance1 = NULL;
86      ALLEGRO_SAMPLE_INSTANCE *instance2 = NULL;
87      ALLEGRO_SAMPLE_INSTANCE *instance3 = NULL;
88
89      // check allegro initialize
90      if(!al_init())
91          return -1;
92
93      // create display
94      display = al_create_display(WIDTH, HEIGHT);
95
96      // check display
97      if(!display)
98          return -1;
```

Berfungsi untuk mendeklarasikan variabel-variabel yang menggunakan allegro, lalu dicek dengan menginisialisasi apakah allegro berjalan lancar. Membuat tampilan awal saat allegro dijalankan.

```
100     al_init_primitives_addon();
101     al_install_keyboard();
102     al_init_font_addon();
103     al_init_ttf_addon();
104     al_init_image_addon();
105     al_install_audio();
106     al_init_acodec_addon();
```

Merupakan parameter yang ada pada allegro

```
108     event_queue = al_create_event_queue();
109     timer = al_create_timer(1.0 / FPS);
110
111     shipImage = al_load_bitmap("spaceship_sprites.png");
112     al_convert_mask_to_alpha(shipImage, al_map_rgb(255, 0, 255));
113
114     cometImage = al_load_bitmap("meteor_sprites.png");
115     al_convert_mask_to_alpha(cometImage, al_map_rgb(0, 0, 0));
116
117     expImage = al_load_bitmap("explosion.png");
118
119     mnImage = al_load_bitmap("mnBg.png");
```

Membuat tampilan games, dengan mengimpor gambar yang telah diconvert agar bisa dijalankan pada allegro.

```
121     srand(time(NULL));
122     InitShip(ship, shipImage);
123     InitBg(Menu, 0, 0, 800, 400, mnImage);
124     InitBullet(bullets, NUM_BULLETS);
125     InitComet(comets, NUM_COMETS, cometImage);
126     InitExplosions(explosions, NUM_EXPLOSIONS, expImage);
```

Mendeklarasikan waktu yang dibutuhkan untuk bermain dengan fungsi random, lalu menginisialisasi objek agar bisa termuat.

```
128     font25 = al_load_font(FONT_DAUNPENH, 25, 0);
129
130     al_reserve_samples(10);
131
132     sample = al_load_sample("horn.ogg");
133     sample2 = al_load_sample("pistol.ogg");
134     sample3 = al_load_sample("gotta.ogg");
135
136     instance1 = al_create_sample_instance(sample);
137     instance2 = al_create_sample_instance(sample2);
138     instance3 = al_create_sample_instance(sample3);
139
140     al_attach_sample_instance_to_mixer(instance1, al_get_default_mixer());
141     al_attach_sample_instance_to_mixer(instance2, al_get_default_mixer());
142     al_attach_sample_instance_to_mixer(instance3, al_get_default_mixer());
143
144     al_register_event_source(event_queue, al_get_keyboard_event_source());
145     al_register_event_source(event_queue, al_get_timer_event_source(timer));
146     al_register_event_source(event_queue, al_get_display_event_source(display));
147
148     al_start_timer(timer);
```

Mendeklarasikan jenis huruf yang akan muncul pada tampilan game sebesar 25pt, lalu juga ada efek suara pada game.

Mendaftarkan sumber-sumber yang telah dideklarasikan tadi seperti waktu dan tampilan layar disesuaikan dengan masukan keyboard.

```
149     while(!done)
150     {
151         ALLEGRO_EVENT ev;
152         al_wait_for_event(event_queue, &ev);
153
154         if(ev.type == ALLEGRO_EVENT_TIMER)
155         {
156             redraw = true;
157             if(keys[UP])
158                 MoveShipUp(ship);
159             else if(keys[DOWN])
160                 MoveShipDown(ship);
161             else
162                 ResetShipAnimation(ship, 1);
163
164             if(keys[LEFT])
165                 MoveShipLeft(ship);
166             else if(keys[RIGHT])
167                 MoveShipRight(ship);
168             else
169                 ResetShipAnimation(ship, 2);
```

Terdapat fungsi while digunakan apabila telah selesai memasukkan sumber-sumber untuk tampilan games, lalu mengatur inputan untuk menggerakkan objek ship, begitu seterusnya sesuai dengan yang dimasukkan user.

```
171             switch(menu){
172             case 1 :
173             break;
174             case 2 :
175                 UpdateExplosions(explosions, NUM_EXPLOSIONS);
176                 UpdateBullet(bullets, NUM_BULLETS);
177                 StartComet(comets, NUM_COMETS);
178                 UpdateComet(comets, NUM_COMETS);
179                 CollideBullet(bullets, NUM_BULLETS, comets, NUM_COMETS, ship, explosions, NUM_EXPLOSIONS);
180                 CollideComet(comets, NUM_COMETS, ship, explosions, NUM_EXPLOSIONS);
181                 if(ship.lives <= 0) {
182                     ship.lives = 5;
183                     menu = 3;
184                 }
185                 al_play_sample_instance(instance2);
186             break;
187             }
188         }
189         else if(ev.type == ALLEGRO_EVENT_DISPLAY_CLOSE)
190         {
191             done = true;
192         }
```

Terdapat pilihan pada menu, saat tampilan sudah berjalan, maka otomatis program akan memulai dengan menampilkan objek-objek dengan memanggil prototype fungsi. done=true untuk mengupdate tampilan. ship.lives digunakan untuk menentukan banyaknya nyawa pada ship/player

```
193         if(ev.type == ALLEGRO_EVENT_KEY_DOWN)
194         {
195             switch(ev.keyboard.keycode)
196             {
197                 case ALLEGRO_KEY_ESCAPE:
198                     done = true;
199                     break;
200                 case ALLEGRO_KEY_UP:
201                     keys[UP] = true;
202                     break;
203                 case ALLEGRO_KEY_DOWN:
204                     keys[DOWN] = true;
205                     break;
206                 case ALLEGRO_KEY_LEFT:
207                     keys[LEFT] = true;
208                     break;
209                 case ALLEGRO_KEY_RIGHT:
210                     keys[RIGHT] = true;
211                     break;
212                 case ALLEGRO_KEY_SPACE:
213                     keys[SPACE] = true;
214                     FireBullet(bullets, NUM_BULLETS, ship);
215                     break;
216                 case ALLEGRO_KEY_ENTER:
217                     if(menu == 1){
218                         menu = 2;
219                     } else {
220                         menu = 1;
221                     }
222                     break;
223             }
224         }
```

Fungsi diatas adalah untuk mengatur masukan keyboard yang nantinya akan digunakan pada games.

```
225         if(ev.type == ALLEGRO_EVENT_KEY_UP)
226         {
227             switch(ev.keyboard.keycode)
228             {
229                 case ALLEGRO_KEY_UP:
230                     keys[UP] = false;
231                     break;
232                 case ALLEGRO_KEY_DOWN:
233                     keys[DOWN] = false;
234                     break;
235                 case ALLEGRO_KEY_LEFT:
236                     keys[LEFT] = false;
237                     break;
238                 case ALLEGRO_KEY_RIGHT:
239                     keys[RIGHT] = false;
240                     break;
241                 case ALLEGRO_KEY_SPACE:
242                     keys[SPACE] = false;
243                     break;
244             }
245         }
```

Funsi diatas digunakan untuk mengatu ship agar tetap pada lintasan/batasan-batasan yang sudah didefinisikan sehingga ship tidak akan berjalan kemana-mana.

```
248         if(redraw && al_is_event_queue_empty(event_queue))
249         {
250             redraw = false;
251             switch(menu){
252             case 1 :
253             DrawBg(Menu);
254             al_play_sample_instance(instance3);
255             break;
```

Berfungsi untuk memuat tampilan utama layar, saat game baru mulai berjalan juga terdapat efek suara yang dimainkan.

```
257         case 2:
258             DrawShip(ship);
259             DrawBullet(bullets, NUM_BULLETS);
260             DrawComet(comets, NUM_COMETS);
261             DrawExplosions(explosions, NUM_EXPLOSIONS);
262 
263             al_draw_textf(font25, al_map_rgb(200, 0, 255), 5, 5, 0, "Player has %i lives left. Player has destroyed %i objects", ship.lives, ship.score);
264             al_draw_textf(font25, al_map_rgb(200, 0, 255), 520, 5, 0, "Press space to shoot");
265             al_draw_textf(font25, al_map_rgb(200, 0, 255), 520, 25, 0, "Press up,down,left,right to move");
266             al_draw_textf(font25, al_map_rgb(200, 0, 255), 10, 380, 0, "Press ESC to exit");
267         break;
268 
269         case 3:
270             al_draw_textf(font25, al_map_rgb(0, 255, 255), 320, 150, 0, "GAME OVER");
271             al_draw_textf(font25, al_map_rgb(0, 255, 255), 320, 200,0, "Final Score  %i.", ship.score);
272             al_draw_textf(font25, al_map_rgb(0, 255, 255), 320, 250, 0, "Press Enter To Retry");
273             al_play_sample_instance(instance1);
274         break;
275         }
276 
277         al_flip_display();
278         al_clear_to_color(al_map_rgb(0, 0, 0));
279     }
```

Fungsi pada case 2 merupakan tampilan saat bermain, sebelumnya juga telah didefinisikan gambar dari objek yang akan ditampilkan. al_draw_textf adalah fungsi dari allegro font untuk

menampilkan karakter huruf pada program, al_map_rgb berfungsi untuk menampilkan warna dari huruf yang ditulis.

Pada case 3 merupakan tampilan saat permainan berakhir.

```
282     al_destroy_bitmap(expImage);
283     al_destroy_bitmap(shipImage);
284     al_destroy_bitmap(cometImage);
285     al_destroy_sample_instance(instance1);
286     al_destroy_sample(sample);
287     al_destroy_sample_instance(instance2);
288     al_destroy_sample(sample2);
289     al_destroy_sample_instance(instance3);
290     al_destroy_sample(sample3);
291     al_destroy_event_queue(event_queue);
292     al_destroy_timer(timer);
293     al_destroy_font(font25);
294     al_destroy_display(display);
295
296     return 0;
297 }
```

Berfungsi untuk melepaskan semua sumber yang telah digunakan, setelah fungsi ini dipanggil maka tidak ada tindakan lainnya, lalu dikembalikan ke nilai 0.

```
299 void InitShip(SpaceShip &ship, ALLEGRO_BITMAP *image)
300 {
301     ship.x = 20;
302     ship.y = HEIGHT / 2;
303     ship.ID = PLAYER;
304     ship.lives = 5;
305     ship.speed = 6;
306     ship.boundx = 10;
307     ship.boundy = 15;
308     ship.score = 0;
309
310     ship.maxFrame = 3;
311     ship.curFrame = 0;
312     ship.frameCount = 0;
313     ship.frameDelay = 50;
314     ship.frameWidth = 44;
315     ship.frameHeight = 44;
316     ship.animationColums = 3;
317     ship.animationDirection = 1;
318
319     ship.animationRow = 1;
320
321     ship.image = image;
322 }
```

Fungsi yang terdapat di luar fungsi utama, digunakan untuk menginisialisasi bentuk dari kapal pada gambar yang telah diatur sedemikian rupa.

```
323  void ResetShipAnimation(SpaceShip &ship, int position)
324  {
325      if(position == 1)
326          ship.animationRow = 1;
327      else
328          ship.curFrame = 0;
329  }
```

Berfungsi untuk mengatur posisi dari ship, apabila ship akan bergerak ke samping maka kapal melakukan pergerakan.

```
330  void DrawShip(SpaceShip &ship) {
331      int fx = (ship.curFrame % ship.animationColums) * ship.frameWidth;
332      int fy = ship.animationRow * ship.frameHeight;
333
334      al_draw_bitmap_region(ship.image, fx, fy, ship.frameWidth,
335          ship.frameHeight, ship.x - ship.frameWidth / 2, ship.y - ship.frameHeight / 2, 0);
336  }
```

Berfungsi untuk mengatur letak ship pada tampilan awal atau saat game baru mulai berjalan.

```
337  void MoveShipUp(SpaceShip &ship)
338  {
339      ship.animationRow = 0;
340      ship.y -= ship.speed;
341      if(ship.y < 0)
342          ship.y = 0;
343  }
```

Berfungsi untuk mengatur pergerakan ship ke atas, maka pergerakannya dibatasi dengan tingginya tampilan yang ada.

```
344  void MoveShipDown(SpaceShip &ship)
345  {
346      ship.animationRow = 2;
347      ship.y += ship.speed;
348      if(ship.y > HEIGHT)
349          ship.y = HEIGHT;
350  }
```

Berfungsi untuk mengatur pergerakan ship ke bawah, maka pergerakannya dibatasi dengan tingginya tampilan yang ada.

```
351  void MoveShipLeft(SpaceShip &ship)
352  {
353      ship.curFrame = 2;
354      ship.x -= ship.speed;
355      if(ship.x < 0)
356          ship.x = 0;
357  }
```

Berfungsi untuk mengatur pergerakan ship ke kiri, maka pergerakannya dibatasi dengan tepi kiri dari lebar tampilan.

```
358  void MoveShipRight(SpaceShip &ship)
359  {
360      ship.curFrame = 1;
361      ship.x += ship.speed;
362      if(ship.x > 300)
363          ship.x = 300;
364  }
```

Berfungsi untuk mengatur pergerakan ship ke kanan, maka pergerakannya dibatasi hanya sejauh 300 pixels dari tepi kiri tampilan, sehingga ship tidak bisa maju sampai batas tepi kanan.

```
366  void InitBullet(Bullet bullet[], int size)
367  {
368      for(int i = 0; i < size; i++)
369      {
370          bullet[i].ID = BULLET;
371          bullet[i].speed = 10;
372          bullet[i].live = false;
373      }
374  }
375  void DrawBullet(Bullet bullet[], int size)
376  {
377      for(int i = 0; i < size; i++)
378      {
379          if(bullet[i].live)
380              al_draw_filled_circle(bullet[i].x, bullet[i].y, 2, al_map_rgb(255, 255, 255));
381      }
382  }
383  void FireBullet(Bullet bullet[], int size, SpaceShip &ship)
384  {
385      for(int i = 0; i < size; i++)
386      {
387          if(!bullet[i].live)
388          {
389              bullet[i].x = ship.x + 17;
390              bullet[i].y = ship.y;
391              bullet[i].live = true;
392              break;
393          }
394      }
395  }
```

Sama seperti fungsi pada objek ship, pertama yang dilakukan adalah menginisialisasi variabel bullet, bullet dideklarasikan saat ship masih bisa bermain, dengan al_map_rgb merupakan parameter untuk membuat warna.

```
396  void UpdateBullet(Bullet bullet[], int size)
397  {
398      for(int i = 0; i < size; i++)
399      {
400          if(bullet[i].live)
401          {
402              bullet[i].x += bullet[i].speed;
403              if(bullet[i].x > WIDTH)
404                  bullet[i].live = false;
405          }
406      }
407  }
```

Berfungsi untuk membuat bullet agar tetap keluar dari ship.

```
void CollideBullet(Bullet bullet[], int bSize, Comet comets[], int cSize, SpaceShip &ship, Explosion explosions[], int eSize)
{
    for(int i = 0; i < bSize; i++)
    {
        if(bullet[i].live)
        {
            for(int j = 0; j < cSize; j++)
            {
                if(comets[j].live)
                {
                    if(bullet[i].x > (comets[j].x - comets[j].boundx) &&
                       bullet[i].x < (comets[j].x + comets[j].boundx) &&
                       bullet[i].y > (comets[j].y - comets[j].boundy) &&
                       bullet[i].y < (comets[j].y + comets[j].boundy))
                    {
                        bullet[i].live = false;
                        comets[j].live = false;

                        ship.score++;

                        StartExplosions(explosions, eSize, bullet[i].x, bullet[i].y);
                    }
                }
            }
        }
    }
}
```

Berfungsi saat bullet mengenai musuh/comets maka akan terjadi ledakan atau efek matinya musuh, juga saat ship bertabrakan dengan musuh/comets.

```
void InitComet(Comet comets[], int size, ALLEGRO_BITMAP *image)
{
    for(int i = 0; i < size; i++)
    {
        comets[i].ID = ENEMY;
        comets[i].live = false;
        comets[i].speed = 3;
        comets[i].boundx = 15;
        comets[i].boundy = 15;

        comets[i].maxFrame = 10;
        comets[i].curFrame = 0;
        comets[i].frameCount = 0;
        comets[i].frameDelay = 5;
        comets[i].frameWidth = 38;
        comets[i].frameHeight = 38;
        comets[i].animationColums = 10;

        comets[i].animationDirection = 1;

        comets[i].image = image;
    }
}
```

Sama seperti fungsi sebelumnya pada masing-masing objek, yaitu ship dan bullet, comets diinisialisasikan terlebih dahulu dengan memuat gambar yang sudah dibuat.

```
459  void DrawComet(Comet comets[], int size)
460  {
461      for(int i = 0; i < size; i++)
462      {
463          if(comets[i].live)
464          {
465              int fx = (comets[i].curFrame % comets[i].animationColums) * comets[i].frameWidth;
466              int fy = (comets[i].curFrame / comets[i].animationColums) * comets[i].frameWidth;
467
468              al_draw_bitmap_region(comets[i].image, fx, fy, comets[i].frameWidth,
469                  comets[i].frameHeight, comets[i].x - comets[i].frameWidth / 2, comets[i].y - comets[i].frameHeight / 2, 0);
470          }
471      }
472  }
473  void StartComet(Comet comets[], int size)
474  {
475      for(int i = 0; i < size; i++)
476      {
477          if(!comets[i].live)
478          {
479              if(rand() % 500 == 0)
480              {
481                  comets[i].live = true;
482                  comets[i].x = WIDTH;
483                  comets[i].y = 30 + rand() % (HEIGHT - 60);
484
485                  break;
486              }
487          }
488      }
489  }
```

Lalu, membuat comets agar bisa masuk ke tampilan awal saat bermain, setelah itu mendeklarasikan pergerakan comets secara random.

```
490  void UpdateComet(Comet comets[], int size)
491  {
492      for(int i = 0; i < size; i++)
493      {
494          if(comets[i].live)
495          {
496              if(++comets[i].frameCount >= comets[i].frameDelay)
497              {
498                  comets[i].curFrame += comets[i].animationDirection;
499                  if(comets[i].curFrame >= comets[i].maxFrame)
500                      comets[i].curFrame = 0;
501                  else if(comets[i].curFrame <= 0)
502                      comets[i].curFrame = comets[i].maxFrame;
503
504                  comets[i].frameCount = 0;
505              }
506
507              comets[i].x -= comets[i].speed;
508
509          }
510      }
511  }
```

Berfungsi untuk mengupdate comets agar tetap muncul selama nyawa dari ship masih ada.

```
void CollideComet(Comet comets[], int cSize, SpaceShip &ship, Explosion explosions[], int eSize)
{
    for(int i = 0; i < cSize; i++)
    {
        if(comets[i].live)
        {
            if(comets[i].x - comets[i].boundx < ship.x + ship.boundx && comets[i].x + comets[i].boundx > ship.x - ship.boundx &&
                comets[i].y - comets[i].boundy < ship.y + ship.boundy && comets[i].y + comets[i].boundy > ship.y - ship.boundy)
            {
                ship.lives--;
                comets[i].live = false;
            }
            else if(comets[i].x < 0)
            {
                comets[i].live = false;
                ship.lives--;
            }

            if(!comets[i].live)
            {
                StartExplosions(explosions, eSize, comets[i].x, comets[i].y);
            }
        }
    }
}
```

Berfungsi untuk mendelarasikan saat nanti ship/player bisa menembak comets/musuh atau saat comets berhasil menyelinap ke kawasan ship maka akan terjadi ledakan.

```
void InitExplosions(Explosion explosions[], int size, ALLEGRO_BITMAP *image)
{
    for(int i = 0; i < size; i++)
    {
        explosions[i].live = false;

        explosions[i].maxFrame = 10;
        explosions[i].curFrame = 0;
        explosions[i].frameCount = 0;
        explosions[i].frameDelay = 1;
        explosions[i].frameWidth = 128;
        explosions[i].frameHeight = 128;
        explosions[i].animationColums = 10;
        explosions[i].animationDirection = 1;

        explosions[i].image = image;

    }
}
```

Sama seperti pada fungsi objek sebelumnya, menginisialisasi objek explosions dengan memuat gambar yang sudah diatur dengan allegro supaya menjadi bitmap dan bisa ditampilkan nantinya.

```
557  void DrawExplosions(Explosion explosions[], int size)
558  {
559      for(int i = 0; i < size; i++)
560      {
561          if(explosions[i].live)
562          {
563              int fx = (explosions[i].curFrame % explosions[i].animationColums) * explosions[i].frameWidth;
564              int fy = (explosions[i].curFrame / explosions[i].animationColums) * explosions[i].frameHeight;
565
566              al_draw_bitmap_region(explosions[i].image, fx, fy, explosions[i].frameWidth,
567                                    explosions[i].frameHeight, explosions[i].x - explosions[i].frameWidth / 2,
568                                    explosions[i].y - explosions[i].frameHeight / 2, 0);
569          }
570      }
571  }
572  void StartExplosions(Explosion explosions[], int size, int x, int y)
573  {
574      for(int i = 0; i < size; i++)
575      {
576          if(!explosions[i].live)
577          {
578              explosions[i].live = true;
579              explosions[i].x = x;
580              explosions[i].y = y;
581
582              break;
583          }
584      }
585  }
```

Berfungsi untuk menempatkan explosions saat player/ship masih hidup, selanjutnya memulai explosions pada posisi x dan y, explosions terjadi saat adanya tabrakan dari masing-masing objek.

```
586  void UpdateExplosions(Explosion explosions[], int size)
587  {
588      for(int i = 0; i < size; i++)
589      {
590          if(explosions[i].live)
591          {
592              if(++explosions[i].frameCount >= explosions[i].frameDelay)
593              {
594                  explosions[i].curFrame += explosions[i].animationDirection;
595                  if(explosions[i].curFrame >= explosions[i].maxFrame)
596                  {
597                      explosions[i].curFrame = 0;
598                      explosions[i].live = false;
599                  }
600
601                  explosions[i].frameCount = 0;
602              }
603          }
604      }
605  }
```

Berfungsi untuk mengupdate explosions agar bisa muncul sewaktu program berjalan.

```
608  void InitBg(Background &bg, int x, int y, int width, int height, ALLEGRO_BITMAP *image)
609  {
610      bg.x = x;
611      bg.y = y;
612      bg.width = width;
613      bg.height = height;
614      bg.image = image;
615  }
616  void DrawBg(Background &bg)
617  {
618      al_draw_bitmap(bg.image, bg.x, bg.y, 0);
619  }
```

Berfungsi menginisialisasi tampilan layar pada program dengan memuat gambar yang sudah diedit, tampilan latar belakang ini akan muncul saat program mulai dijalankan untuk memilih menu pada game.

ANALISIS JALANNYA PROGRAM

Cara bermain untuk game spacy ini sangat mudah, berikut tombol yang digunakan untuk bermain :

- Enter : untuk memulai game
- space : untuk menembak
- → : untuk bergeser ke kanan
- ← : untuk bergeser ke kiri
- ↓ : untuk bergeser ke bawah
- ↑ : untuk bergeser ke atas
- Esc : untuk keluar dari game

Pada game int terdapat pemain berupa pesawat dan musuh berupa komet, pemain memiliki 5 nyawa untuk menembak musuh, tidak diberikan batasan waktu, pemain diharapkan bisa menembak sebanyak-banyaknya.

Berikut adalah tampilan dari game spacy :


games.exe
Player has 4 lives left. Player has destroyed 17 objects
Press space to shoot
Press up,down,left,right to move
Press ESC to exit


games.exe
GAME OVER
Final Score 0.
Press Enter To Retry