BINUS UNIVERSITY
BINUS INTERNATIONAL

Assignment Cover Letter

(Individual Work)

Student Information:	**Surname**	**Given Names**	**Student ID Number**
1.	**Sandjaya**	**Guntur**	**2101684563**

Course Code	**: COMP6502**	**Course Name**	**: Introduction to Programming**
Class	**: L1AC**	**Name of Lecturer(s)**	**:** 1. Bagus Kerthyayana 2. Tri Asih Budiono
Major	**: CS**		
Title of Assignment (if any)	: **Duck Hunt (Game)**		
Type of Assignment	**: Final Project**		
Submission Pattern			
Due Date	**: 6-11-2017**	**Submission Date**	**: 6-11-2017**

The assignment should meet the below requirements.

1. Assignment (hard copy) is required to be submitted on clean paper, and (soft copy) as per lecturer's instructions.
2. Soft copy assignment also requires the signed (hardcopy) submission of this form, which automatically validates the softcopy submission.
3. The above information is complete and legible.
4. Compiled pages are firmly stapled.
5. Assignment has been copied (soft copy and hard copy) for each student ahead of the submission.

Plagiarism/Cheating

BiNus International seriously regards all forms of plagiarism, cheating and collusion as academic offenses which may result in severe penalties, including loss/drop of marks, course/class discontinuity and other possible penalties executed by the university. Please refer to the related course syllabus for further information.

Declaration of Originality

By signing this assignment, Iunderstand, accept and consent to BiNus International terms and policy on plagiarism. Herewith Ideclare that the work contained in this assignment is myown work and has not been submitted for the use of assessment in another course or class, except where this has been notified and accepted in advance.

Signature of Student:

(Name of Student)
Guntur Sandjaya

“Duck Hunt”

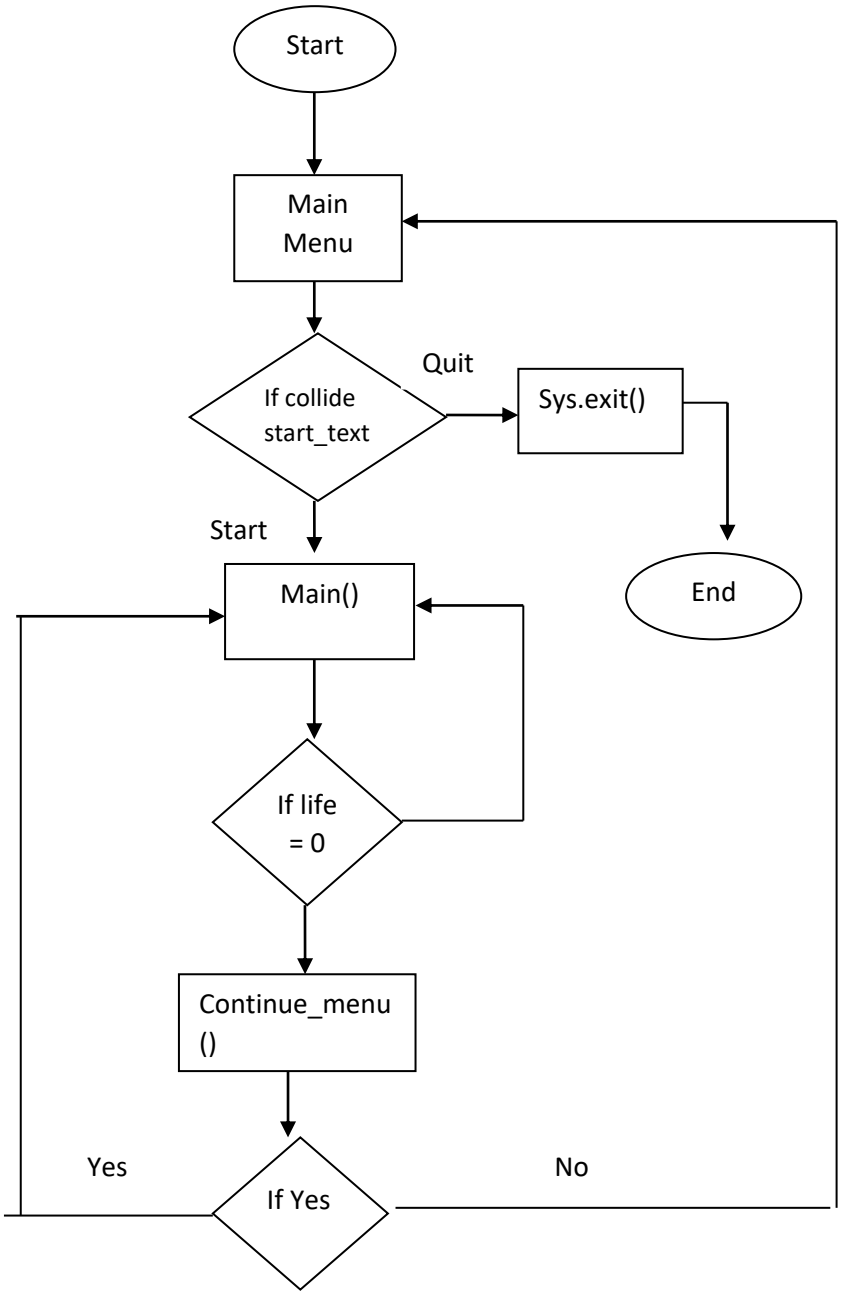
Name : Guntur Sandjaya

ID : 2101684563

I. Description

This game is inspired from Duck Hunt in Nintendo. The objective of this game is getting high score by shooting the duck as much as possible. The purpose of creating this game is to fulfill my assignment for final project. The reason that I choose this game as final project assignment is to remind the game that exists in my childhood.

II. Flow Chart



III. Explanation of the function

1. Class Bird()

   Def __init__(self): , to initialize the attribute of the class. In here I create an image of bird which is the part of sprite library so I can generate it. To call the image I use pygame.image.load . to get the hit box I use self.image.getrect(). To make It randomly start I use self.rect.y = random.randint(100,400).

   Def flee(self), to move the image I use self.rect.right += random.randint(1,5)*4. So it will move to the right in the speed in random 1x4 or 5x1.

   Def Draw(self.screen): to draw the image on the screen, using screen.blit(self.image, self.rect)

2. Class Aim()

   Def __init__(self): , to initialize the attribute of the class. In here I create an image of bird which is the part of sprite library so I can generate it. To call the image I use pygame.image.load . to get the hit box I use self.image.getrect().

   Def hit(self, target): if it collide with target the it will return the value.

   Def update(self,pt), it will be in the center with the pointer.

   Def draw(self,screen), to draw the image on the screen, using screen.blit(self.image, self.rect)

3. Text_Menu(Sprite):

   Def __init__(self): , to initialize the attribute of the class. In here I create an image of bird which is the part of sprite library so I can generate it. In here the intial attribute is the fontsize, fontstyle, text (the word that want we input), x position, y position, and RGB (the colour). I also convert this text into image by using self.font.render so it will appear in the screen. Self.rect so it can be collide with the pointer / cursor.

4. Main variable

```
# ---------- Main ----------------------
pygame.init()
screen = pygame.display.set_mode((1080,729))
pygame.display.set_caption('Duck Hunt')
background = pygame.image.load('stage.png')
scrWidth, scrHeight = screen.get_size()
mousepos = (scrWidth/2, scrHeight/2)
gun_snd = pygame.mixer.Sound('Duck Hunt SFX (13).wav')
gun_snd.set_volume(1)
```

These are the main variable that will be use in the def main(): . Screen is used to create the user interface screen. Displaye set caption to create the title of the game. Background to create a background image. scrWidth, scrHeight get the size of the user interface screen. Mousepos to get the mouse position. Gun_snd to insert sound. Gun_snd.set_volume to set the volume of the sound.

5. Def main():

```
# Main program / How the game works
def main():

    clock = pygame.time.Clock()
    pygame.mouse.set_visible(False)
    mousepos = (scrWidth/2, scrHeight/2)
    running = True
    bird_timer = pygame.time.get_ticks()
    bird_group = Group()
    aim = Aim()
    score = 0
    life = 5
```

These are the variable that will be used in the game. Clock to get the frame per second. Set the moause visible false so it will not appear will in game. Running = True is a variable for the while loop. Bird timer to get time now. Bird group to create a group of the bird. Score and life Is variable for the score and life.

```
while running:
    text_score = Text_Menu(40,'Times New Roman','Score :' + str(score), 0, 0, 255, 255, 255)
    text_life = Text_Menu(40,'Times New Roman','Life :' + str(life), 0,80, 255, 255, 255)
    text_appear = Group(text_score,text_life)

    clock.tick(60)
```

While running, text_score is the variable to create the score text in size 40, font: times new roman, in position 0, 0. text_life to create text life in size 40m, font: Times New Roman, in position 0, 80. Text_appear is a group for the text_score and text_life. Clocktick is for the frame per second.

```
# To summon the bird
        if pygame.time.get_ticks() - bird_timer >= 3000:
            bird = Bird()
            bird_group.add(bird)
            bird_timer = pygame.time.get_ticks()
```

If this code is executed every 3000 millisecond it will create a bird and it will be add in the bird group.

```
# how the bird work in the game
        for bird in bird_group:
            bird.flee()
            if bird.rect.left >= 1200:
                bird_group.remove(bird)
                life -= 1
            if life == 0:
                running = False
                mouse.set_visible(True)
                continue_menu(score)
```

If this code I executed every bird in bird group will flee. If the bird move more than 1200, remove the bird and life - 1. if life = 0 running will false then it will go to the continue menu

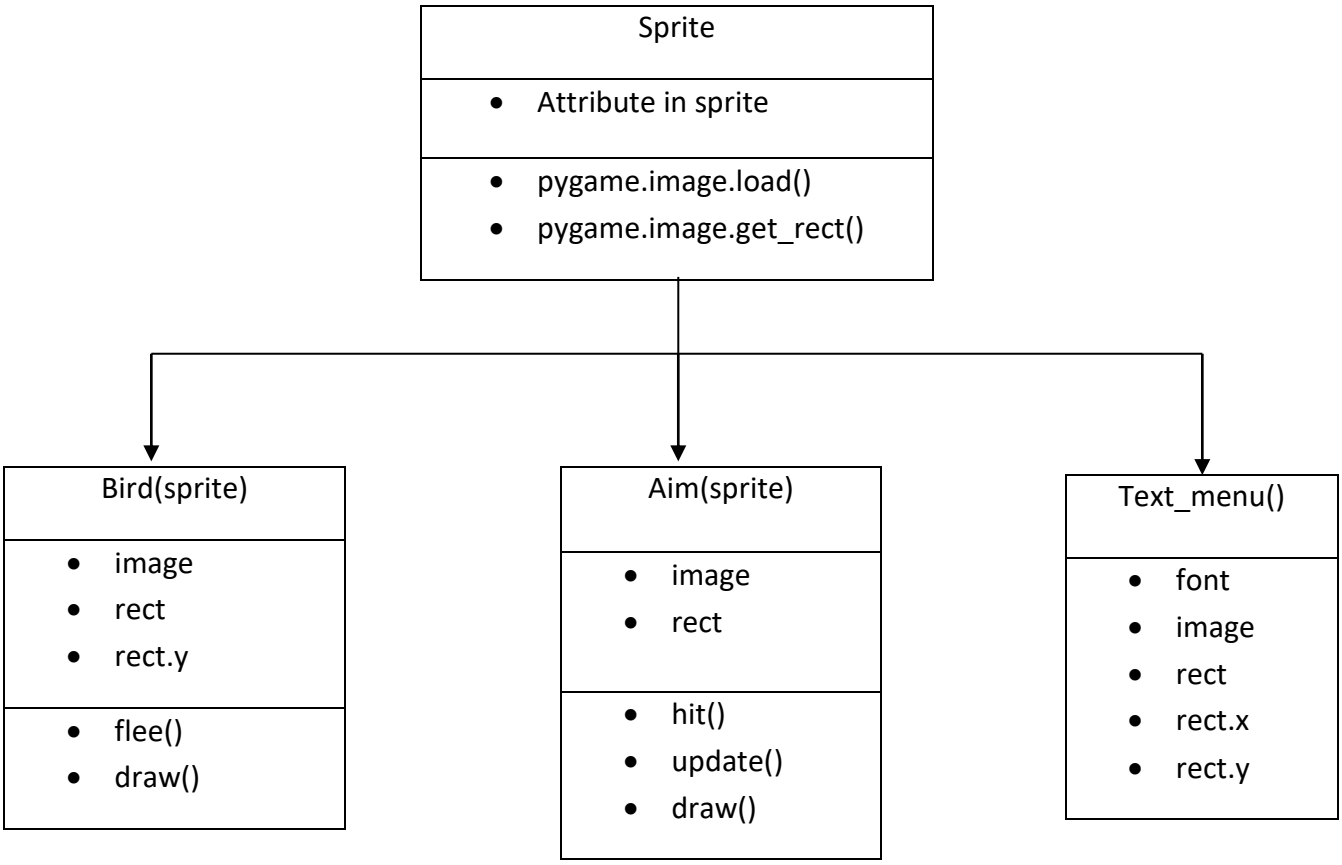
```
# condition in the game
        for event in pygame.event.get():
            if event.type == QUIT:
                exit()
            if event.type == MOUSEMOTION:
                mousepos = pygame.mouse.get_pos()
            if event.type == MOUSEBUTTONDOWN:
                running = True
                gun_snd.play()
                if spritecollide(aim, bird_group, dokill=True):
                    score += 100

                else:
                    life -= 1
                    if life == 0:
                        running = False
                        mouse.set_visible(True)
                        continue_menu(score)


        aim.update(mousepos)
```

For every event in the game if it quit the game will be exit. If it mouse motion it will go to the mouse position. If mouse button down it will still running and gun sound will be play and if aim collide with the bird group do kill and score + 100. Else life – 1 and if life = 0 running will false and it will go to continue menu. Aim.update to update the mouse with aim in position mousepos.

IV. UML Diagram



V. Code

VI.

```
import pygame
import sys
import random
from pygame import *
from pygame.sprite import *
from pygame.sysfont import *



# ----------------------------------------
class Bird(pygame.sprite.Sprite):
    def __init__(self):
        Sprite.__init__(self)
        self.image = pygame.image.load('s-l500.png')
        self.rect = self.image.get_rect()
        self.rect.y = random.randint(100,400)

# bird move to the right
    def flee(self):
        self.rect.right += random.randint(1,5)*4



    def draw(self,screen):
        screen.blit(self.image, self.rect)

# ----------------------------------------

class Aim(pygame.sprite.Sprite):

    def __init__(self):
        Sprite.__init__(self)
        self.image = pygame.image.load('target_PNG8.png')
        self.rect = self.image.get_rect()

    def hit(self,target):
        return self.rect.colliderect(target)

    def update(self,pt):
        self.rect.center = pt

    def draw(self,screen):
        screen.blit(self.image, self.rect)

# ----------------------------------------
class Text_Menu(Sprite):
    def __init__(self,fontsize, fontstyle, text, xpos, ypos, R, G, B):
        Sprite.__init__(self)
        self.font = pygame.font.SysFont(fontstyle,fontsize)
        self.image = self.font.render(text, False, (R,G,B))
        self.rect = self.image.get_rect()
        self.rect.x = xpos
        self.rect.y = ypos


# ---------- Main ----------------------
pygame.init()
screen = pygame.display.set_mode((1080,729))
pygame.display.set_caption('Duck Hunt')
background = pygame.image.load('stage.png')
scrWidth, scrHeight = screen.get_size()
mousepos = (scrWidth/2, scrHeight/2)
gun_snd = pygame.mixer.Sound('Duck Hunt SFX (13).wav')
gun_snd.set_volume(1)
```

```
# ------------------------------------------------------------
# Main program / How the game works
def main():

    clock = pygame.time.Clock()
    pygame.mouse.set_visible(False)
    mousepos = (scrWidth/2, scrHeight/2)
    bird_timer = pygame.time.get_ticks()
    bird_group = Group()
    aim = Aim()
    score = 0
    life = 5
    running = True


    while running:
        text_score = Text_Menu(40,'Times New Roman','Score :' + str(score),
0, 0, 255, 255, 255)
        text_life = Text_Menu(40,'Times New Roman','Life :' + str(life),
0,80 , 255, 255, 255)
        text_appear = Group(text_score,text_life)

        clock.tick(60)

# To summon the bird
        if pygame.time.get_ticks() - bird_timer >= 3000:
            bird = Bird()
            bird_group.add(bird)
            bird_timer = pygame.time.get_ticks()

# how the bird work in the game
        for bird in bird_group:
            bird.flee()
            if bird.rect.left >= 1200:
                bird_group.remove(bird)
                life -= 1
            if life == 0:
                running = False
                mouse.set_visible(True)
                continue_menu(score)

# condition in the game
        for event in pygame.event.get():
            if event.type == QUIT:
                exit()
            if event.type == MOUSEMOTION:
                mousepos = pygame.mouse.get_pos()
            if event.type == MOUSEBUTTONDOWN:
                running = True
                gun_snd.play()
                if spritecollide(aim, bird_group, dokill=True):
                    score += 100

                else:
                    life -= 1
                    if life == 0:
                        running = False
                        mouse.set_visible(True)
                        continue_menu(score)


        aim.update(mousepos)


# draw / update the picture
        screen.blit(background, (0,0))
        bird_group.draw(screen)
        aim.draw(screen)
```

```
            text_appear.draw(screen)

            pygame.display.update()

    # ----------------------------------------------------
    # Continue Menu when the life = 0
    def continue_menu(score):
        bg = pygame.image.load('stage.png')

        pygame.mouse.set_visible(True)
        screen = pygame.display.set_mode((1280,729))
        text = Text_Menu(100,"Times New Roman", "Continue",  450,200 , 255, 255,
    255)
        text_yes = Text_Menu(80,'Times New Roman','Yes', 470,300,255,255,255)
        text_no = Text_Menu(80,'Times New Roman','No',700,300,255,255,255)
        text_score = Text_Menu(80,'Times New Roman','Total Score :'+
    str(score),450,100,255,255,255)
        screen.blit(bg,(0,0))


        while True:

            screen.blit(bg,(0,0))

            text_appear = Group(text,text_yes,text_no,text_score)
            text_appear.draw(screen)



            e = event.wait()
            if text_yes.rect.collidepoint(mouse.get_pos()):
                if e.type == MOUSEBUTTONDOWN:
                    main()
            if text_no.rect.collidepoint(mouse.get_pos()):
                if e.type == MOUSEBUTTONDOWN:
                    menu()
                    break
            if e.type == QUIT:
                sys.exit()

            display.update()


    # ----------------------------------------
    # Main Menu
    def menu():
        bg = pygame.image.load('stage.png')

        pygame.mouse.set_visible(True)
        screen = pygame.display.set_mode((1280,729))
        pygame.display.set_caption('Duck Hunt')
        text_start = Text_Menu(40,"Times New Roman", "Play",  100, 100, 255,
    255, 255)
        text_exit = Text_Menu(40,"Times New Roman",'Quit', 100, 200, 255, 255,
    255)
        screen.blit(bg,(0,0))


        while True:

            screen.blit(bg,(0,0))

            text_appear = Group(text_start,text_exit)
            text_appear.draw(screen)
            title = pygame.image.load('Duck-Hunt.png')
            screen.blit(title,(500, 300))

            e = event.wait()
```

```
            if text_start.rect.collidepoint(mouse.get_pos()):
                if e.type == MOUSEBUTTONDOWN:
                    main()
            if text_exit.rect.collidepoint(mouse.get_pos()):
                if e.type == MOUSEBUTTONDOWN:
                    sys.exit()
            if e.type == QUIT:
                pygame.quit
                break
            display.update()

    menu()

    # Special Thanks to Dumac, Pier, Georgius, and Excelino
    # Sound from https://www.sounds-resource.com/nes/duckhunt/sound/4233/
```