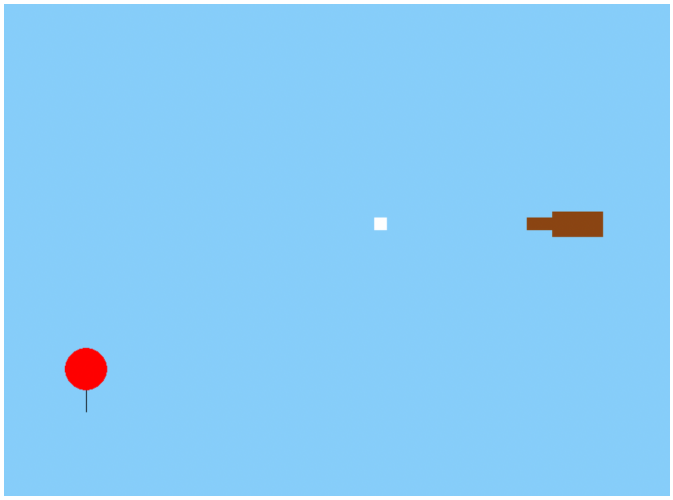
Balloon Shooting Game: Student Manual

Neysan Foo

1 Introduction

In this instructional manual, we will guide you through the process of creating a simple Balloon Shooting Game using Python and the Pygame library. The game consists of a cannon that can be moved up and down using the arrow keys, and a balloon that moves randomly. The player can shoot bullets by pressing the space key. The goal is to shoot the balloon down. We will show the player how many shots were missed at the end.



2 Requirements

Before you start, make sure you have the following software installed on your computer:

- Python 3.x
- Pygame library

To install the Pygame library, open a terminal or command prompt and run:

```
pip install pygame
```

3 Getting Started

First, create a new Python file (e.g., `"balloon_shooting_game.py"`) and open it in your favorite text editor or integrated development environment (IDE). Then, import the required modules at the beginning of the file:

```
import pygame
import sys
import random
```

4 Setting up the Environment

Now, let's set up the game environment, including initializing Pygame, defining constants, creating a display surface, and initializing the game clock.

4.1 Initialize Pygame

Add the following line to initialize the Pygame library:

```
pygame.init()
```

4.2 Define Constants

Define the necessary constants for your game, such as the screen dimensions, colors, and font colors:

```
WIDTH, HEIGHT = 800, 600
BG_COLOR = (135, 206, 250)
CANNON_COLOR = (139, 69, 19)
BALLOON_COLOR = (255, 0, 0)
BULLET_COLOR = (255, 255, 255)
FONT_COLOR = (0, 0, 0)
```

4.3 Create Display Surface

Create a display surface and set the window title:

```
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Balloon␣Shooting␣Game")
```

4.4 Initialize Game Clock

Add a game clock to control the game's frame rate:

```
clock = pygame.time.Clock()
```

4.5 Load Font

Load the font for displaying text, such as countdown and score:

```
font = pygame.font.Font(None, 36)
```

5 Creating Game Objects

In this section, we will create three classes for our game: Cannon, Balloon, and Bullet. We will provide you with the starter code, and you will need to complete the implementation of the classes by following the instructions provided in the comments.

5.1 Cannon Class

Create a Cannon class with an initializer and a draw method:

```
class Cannon:
    def __init__(self, x, y, width, height):
        self.x = x
        self.y = y
        self.width = width
        self.height = height

    def draw(self, screen):
        pygame.draw.rect(screen, CANNON_COLOR, (self.x, self.y, self.width, self.
            height))
        pygame.draw.rect(screen, CANNON_COLOR, (self.x - self.width // 2, self.y
            + self.height // 4, self.width // 2, self.height // 2))
```

5.2 Balloon Class

Create a Balloon class with an initializer, a move method, and a draw method:

```
class Balloon:
    def __init__(self, x, y, radius):
        self.x = x
        self.y = y
        self.radius = radius
        self.speed = 1
        self.direction = random.choice([-1, 1])

    def draw(self, screen):
        pygame.draw.circle(screen, BALLOON_COLOR, (self.x, self.y), self.radius)
        pygame.draw.aaline(screen, (0, 0, 0), (self.x, self.y + self.radius), (
            self.x, self.y + self.radius * 2))

    def move(self):
        # ... (balloon movement code)
        # Update the Balloon's position based on its current speed and direction
```

5.3 Bullet Class

Create a Bullet class with an initializer, a move method, and a draw method:

```
class Bullet:
    def __init__(self, x, y, height):
        self.x = x
        self.y = y
        self.height = height
        self.speed = -10

    def draw(self, screen):
        pygame.draw.rect(screen, BULLET_COLOR, (self.x, self.y, self.height, self
            .height))

    def move(self):
        # ... (bullet movement code)
        # Update the Bullet's position based on its current speed
```

6 Main Game Loop

TODO

7 Display Results and Exit Game

TODO