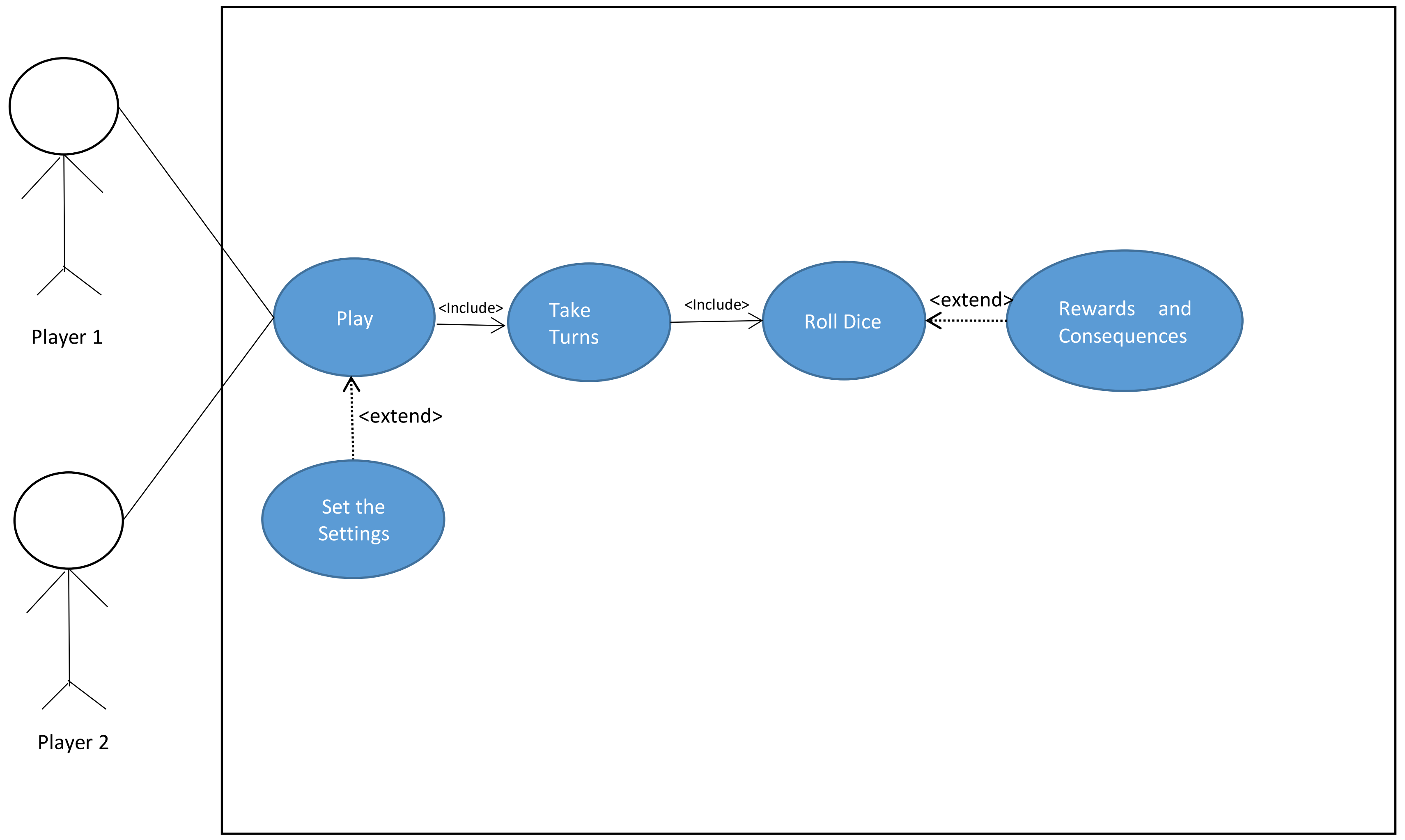
Project Documentations
DailyBasis

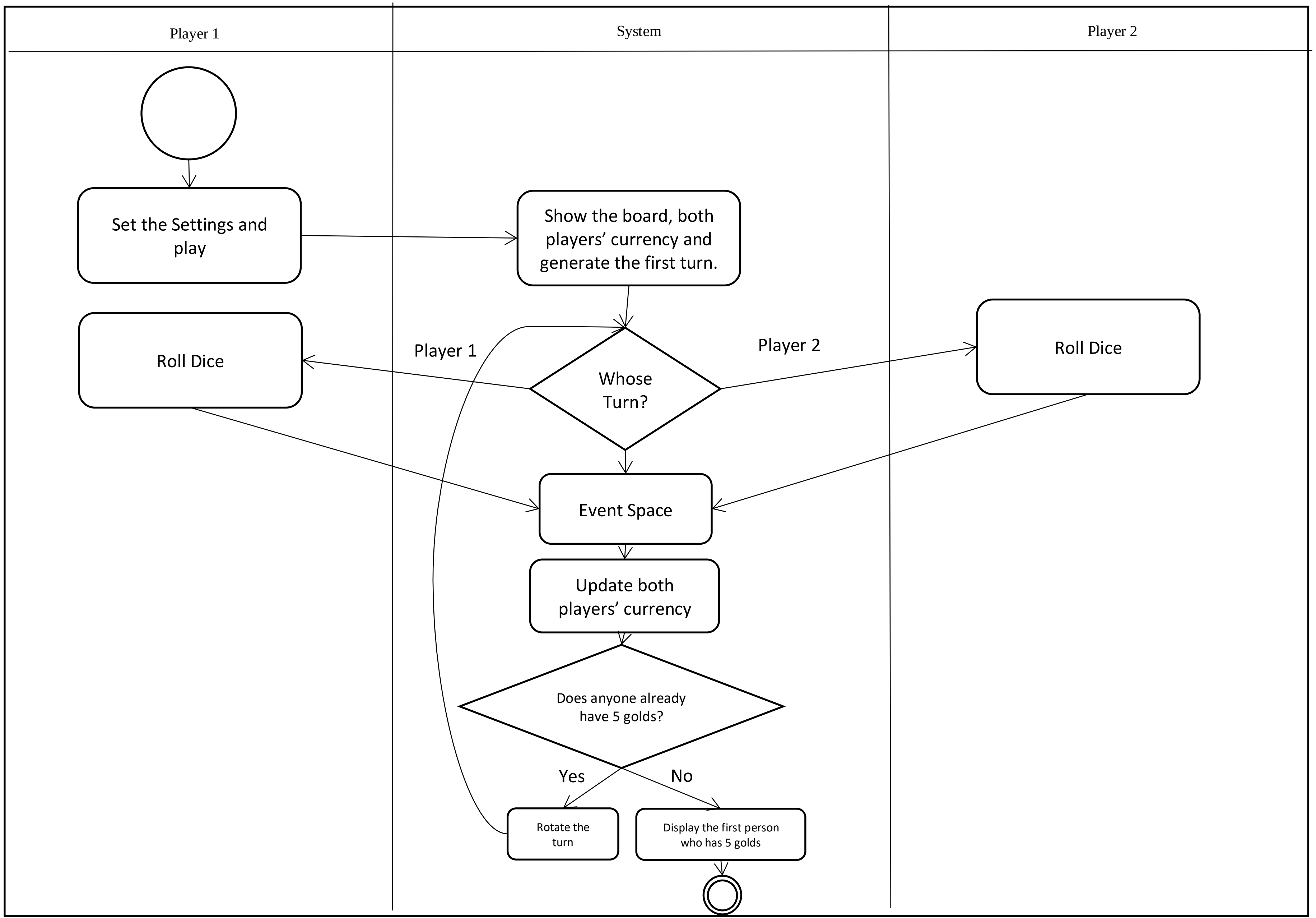
A. Brief Descriptions

Have you ever heard a game that is only required your luckiness and only a little skill to beat your opponents? Mostly, this type of game are in board games, such as *Mario Party*, *Monopoly and Snakes and Ladders*. Furthermore, when the players play those types of games especially with their friends, it can ruin their friendship in a good way. Not only they are ruining their friendship by playing it, but also they are having more fun. With the combination of three games references that have been mentioned, here's the DailyBasis come in to destroy your friendship. The goal of this game is that whoever has five golds win the game.

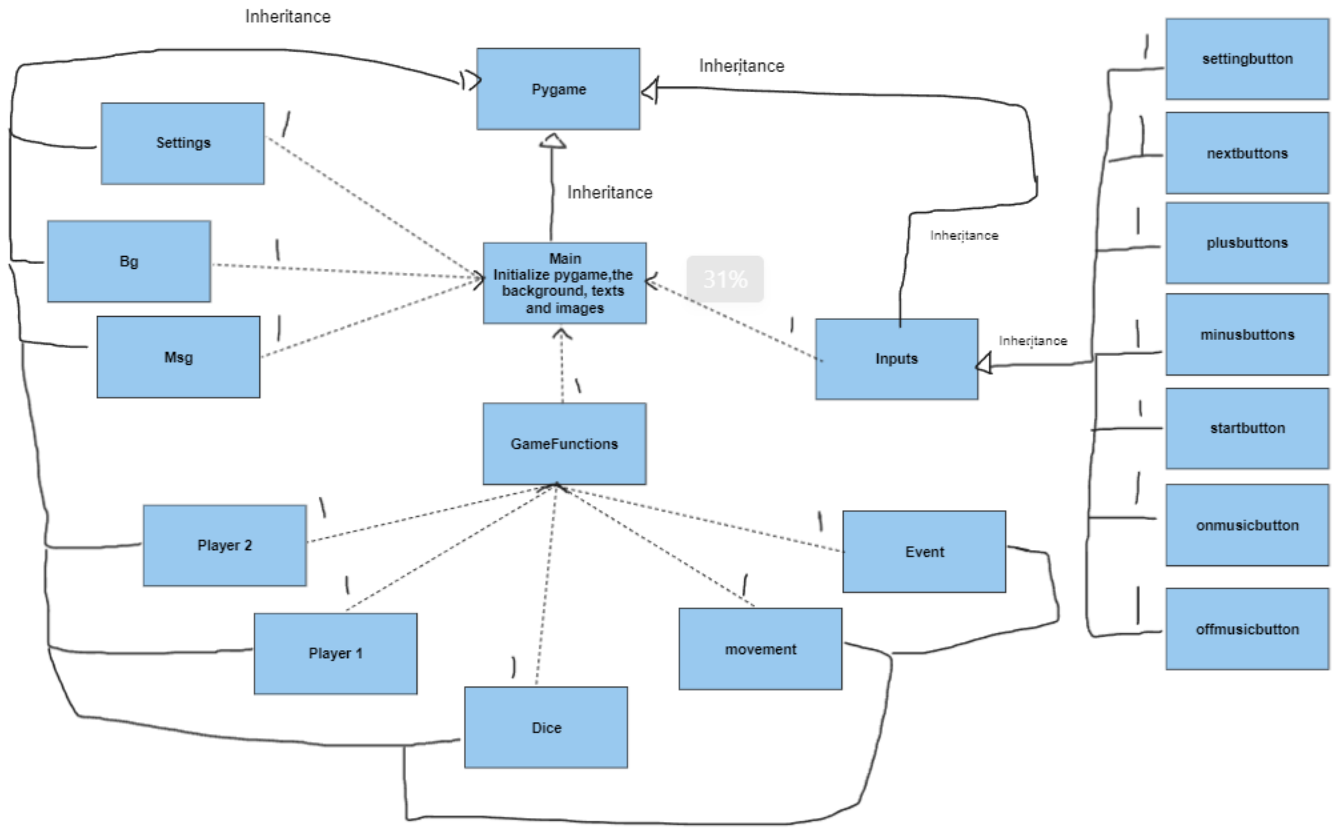
B. Use case diagram



C. Activity Diagram


Player 1
System
Player 2
Set the Settings and play
Show the board, both players' currency and generate the first turn.
Roll Dice
Player 1
Whose Turn?
Player 2
Roll Dice
Event Space
Update both players' currency
Does anyone already have 5 golds?
Yes
No
Rotate the turn
Display the first person who has 5 golds

D. Class Diagram



E. Modules

Here are the modules that I used in the program:

- PyGame

Pygame is one of the best modules for creating a video game. This is because they provide features that will be used in the video games such as buttons, GUI, and others. Since it is a great modules to create a game. Thus, most of the codes were using the pygame codes. For example, pygame.font.sysFont and pygame.font.render, they are used to display texts. Then, pygame.display.flip is used for updating the whole screen. Other than that, pygame.mouse.get_pos is used to get the position of the mouse and BPos.collidepoint to register the button clicks and more.

- Sys

By using sys modules, we can use it for ending the program. Moreover, it has a relevance with pygame. Whenever pygame needs to render a font, pygame can use the SysFont to render available sys modules' font.

- Time

Time modules is really helpful when it comes to delay a certain event. For example, if the player reached the "chance" space, the back side of the chance card will appear first and last for three seconds. Then, the front side of the chance card will appear and last for three seconds as well. Lastly, the updated currency will be displayed. In here, the programmer using the time modules to make the chance card appears longer time.

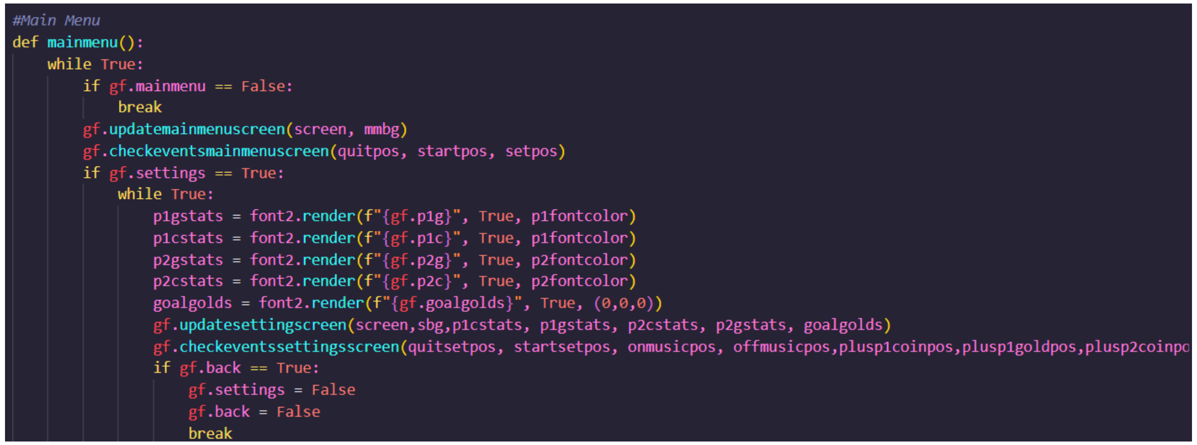
- Random

Based on its name, this module will generate a random number and even randomly choose a variable from a list. In here, the programmer used it on first turn function to determine who goes first. Furthermore, the random numbers are used to choose the front side of “chance” or “badluck” or “gacha” card and roll a dice.

F. Essential Algorithms

A. Main Menu and Settings

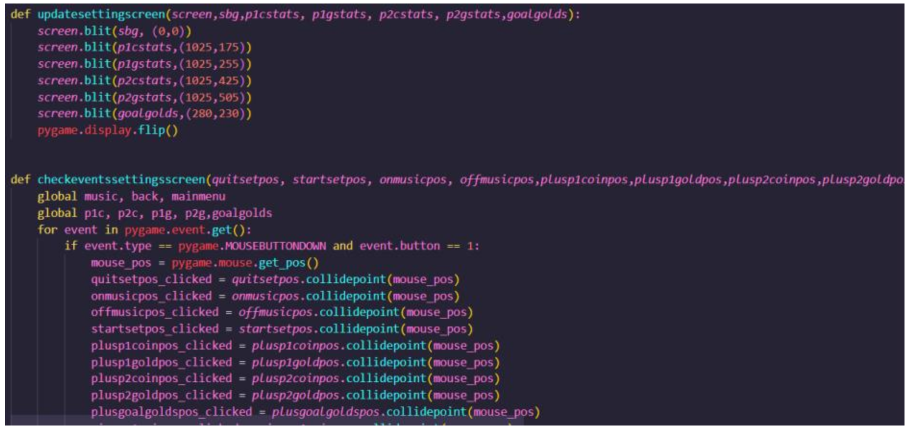
Firstly, when the user launch the game, the user will be welcomed with the main menu where it displays the logo, “quit” button (quit the game), “play” button (play the game) and also the “settings” button (customize the game settings and general settings).



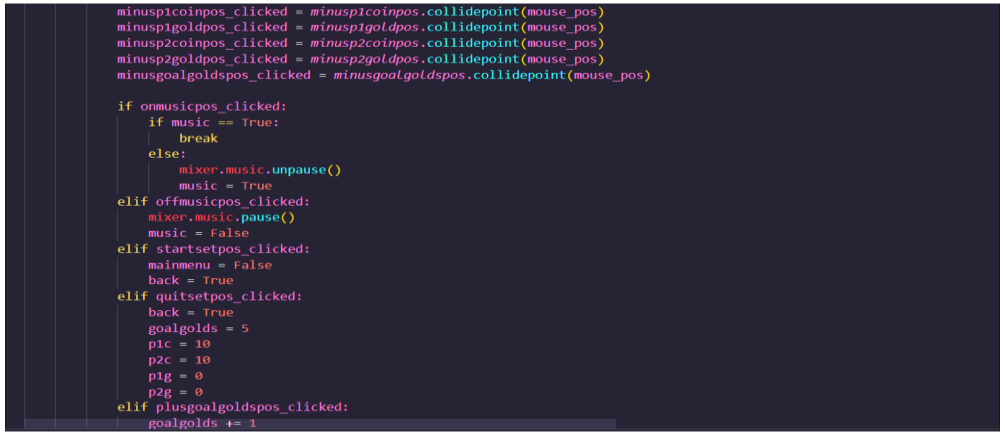
```python
#Main Menu
def mainmenu():
    while True:
        if gf.mainmenu == False:
            break
        gf.updatemainmenuscreen(screen, mmbg)
        gf.checkeventsmainmenuscreen(quitpos, startpos, setpos)
        if gf.settings == True:
            while True:
                p1gstats = font2.render(f"{gf.p1g}", True, p1fontcolor)
                p1cstats = font2.render(f"{gf.p1c}", True, p1fontcolor)
                p2gstats = font2.render(f"{gf.p2g}", True, p2fontcolor)
                p2cstats = font2.render(f"{gf.p2c}", True, p2fontcolor)
                goalgolds = font2.render(f"{gf.goalgolds}", True, (0,0,0))
                gf.updatesettingscreen(screen,sbg,p1cstats, p1gstats, p2cstats, p2gstats, goalgolds)
                gf.checkeventssettingsscreen(quitsetpos, startsetpos, onmusicpos, offmusicpos,plusp1coinpos,plusp1goldpos,plusp2coinpo
                if gf.back == True:
                    gf.settings = False
                    gf.back = False
                    break
```

Picture 1.1: Main Menu Looping Code (main.py)

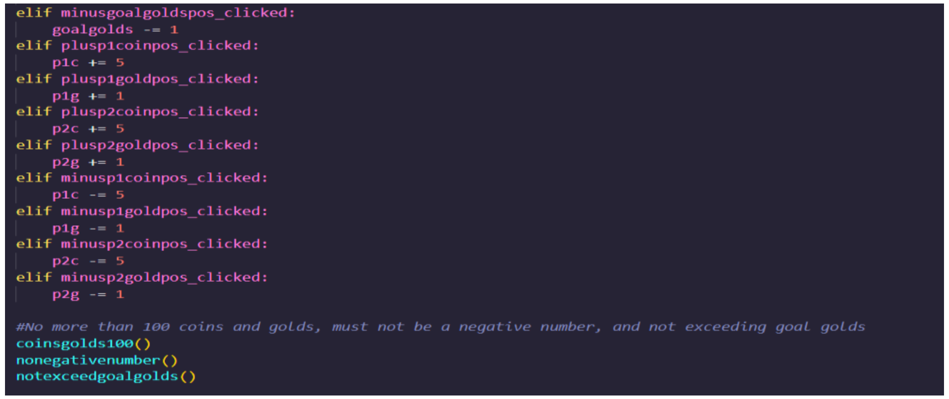
As we can see in the picture above, mainmenu() functions receive inputs (gf.checkeventsmainmenuscreen) and also update the display for the main menu (gf.updatemainmenuscreen). If the users click the settings button in the main menu, the while loop for settings will be occurred which will render the stats for each player and the win condition, and also the settings screen overlaps the main menu screen.



```python
def updatesettingscreen(screen,sbg,p1cstats, p1gstats, p2cstats, p2gstats,goalgolds):
    screen.blit(sbg, (0,0))
    screen.blit(p1cstats,(1025,175))
    screen.blit(p1gstats,(1025,255))
    screen.blit(p2cstats,(1025,425))
    screen.blit(p2gstats,(1025,505))
    screen.blit(goalgolds,(280,230))
    pygame.display.flip()


def checkeventssettingsscreen(quitsetpos, startsetpos, onmusicpos, offmusicpos,plusp1coinpos,plusp1goldpos,plusp2coinpos,plusp2goldpo
    global music, back, mainmenu
    global p1c, p2c, p1g, p2g,goalgolds
    for event in pygame.event.get():
        if event.type == pygame.MOUSEBUTTONDOWN and event.button == 1:
            mouse_pos = pygame.mouse.get_pos()
            quitsetpos_clicked = quitsetpos.collidepoint(mouse_pos)
            onmusicpos_clicked = onmusicpos.collidepoint(mouse_pos)
            offmusicpos_clicked = offmusicpos.collidepoint(mouse_pos)
            startsetpos_clicked = startsetpos.collidepoint(mouse_pos)
            plusp1coinpos_clicked = plusp1coinpos.collidepoint(mouse_pos)
            plusp1goldpos_clicked = plusp1goldpos.collidepoint(mouse_pos)
            plusp2coinpos_clicked = plusp2coinpos.collidepoint(mouse_pos)
            plusp2goldpos_clicked = plusp2goldpos.collidepoint(mouse_pos)
            plusgoalgoldspos_clicked = plusgoalgoldspos.collidepoint(mouse_pos)
```



```python
            minusp1coinpos_clicked = minusp1coinpos.collidepoint(mouse_pos)
            minusp1goldpos_clicked = minusp1goldpos.collidepoint(mouse_pos)
            minusp2coinpos_clicked = minusp2coinpos.collidepoint(mouse_pos)
            minusp2goldpos_clicked = minusp2goldpos.collidepoint(mouse_pos)
            minusgoalgoldspos_clicked = minusgoalgoldspos.collidepoint(mouse_pos)

            if onmusicpos_clicked:
                if music == True:
                    break
                else:
                    mixer.music.unpause()
                    music = True
            elif offmusicpos_clicked:
                mixer.music.pause()
                music = False
            elif startsetpos_clicked:
                mainmenu = False
                back = True
            elif quitsetpos_clicked:
                back = True
                goalgolds = 5
                p1c = 10
                p2c = 10
                p1g = 0
                p2g = 0
            elif plusgoalgoldspos_clicked:
                goalgolds += 1
```

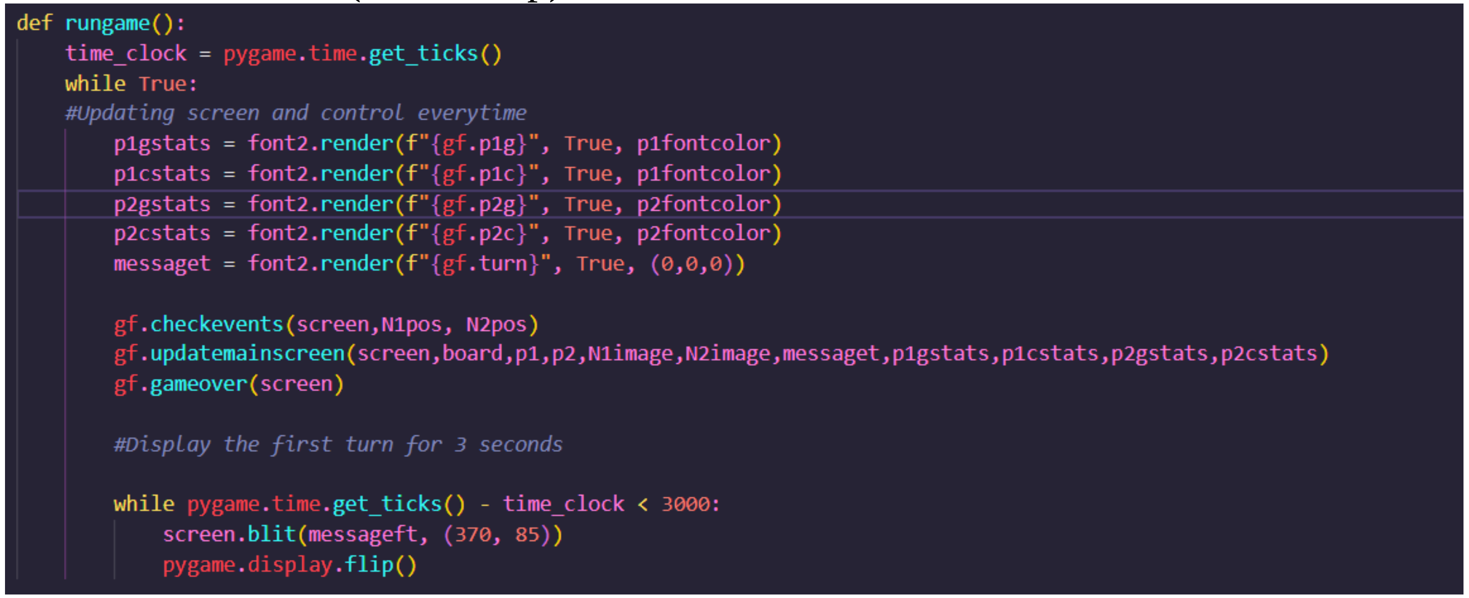


```python
elif minusgoalgoldspos_clicked:
    goalgolds -= 1
elif plusp1coinpos_clicked:
    p1c += 5
elif plusp1goldpos_clicked:
    p1g += 1
elif plusp2coinpos_clicked:
    p2c += 5
elif plusp2goldpos_clicked:
    p2g += 1
elif minusp1coinpos_clicked:
    p1c -= 5
elif minusp1goldpos_clicked:
    p1g -= 1
elif minusp2coinpos_clicked:
    p2c -= 5
elif minusp2goldpos_clicked:
    p2g -= 1

#No more than 100 coins and golds, must not be a negative number, and not exceeding goal golds
coinsgolds100()
nonegativenumber()
notexceedgoalgolds()
```

Picture 1.2, 1.3, 1.4: Settings Functions (gamefunction.py)

In the settings screen, there are a lot of buttons that can be clicked. Starting off with the basic one, the "quit" button which will remove the settings display from the screen and automatically back to the main menu screen. In the game settings area, there are two buttons available and a number that displays the goalgolds value. The red buttons will substract one goalgolds value, however the green button will add one goalgolds value. Below game settings, there is general settings that contains "toggle music" where you can switch the music on or off. In here, the program uses pause method. Thus, when the music is turned on again, the music will be resumed. In the upper right screen, there is handicap area which the users can set the initial coin and gold value for each player. There are also some extra functions which makes the system does not break. The extra functions will be discussed after main game functions.

B. The Main Game (Game loop)

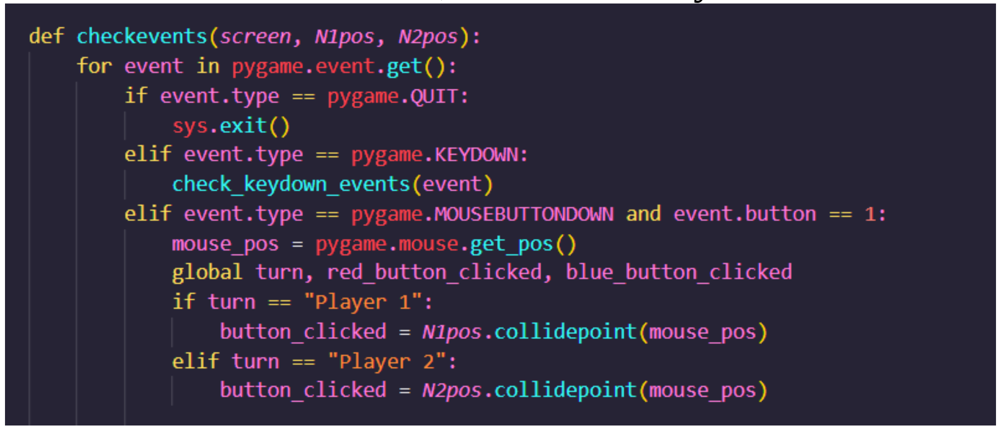


```python
def rungame():
    time_clock = pygame.time.get_ticks()
    while True:
    #Updating screen and control everytime
        p1gstats = font2.render(f"{gf.p1g}", True, p1fontcolor)
        p1cstats = font2.render(f"{gf.p1c}", True, p1fontcolor)
        p2gstats = font2.render(f"{gf.p2g}", True, p2fontcolor)
        p2cstats = font2.render(f"{gf.p2c}", True, p2fontcolor)
        messaget = font2.render(f"{gf.turn}", True, (0,0,0))

        gf.checkevents(screen,N1pos, N2pos)
        gf.updatemainscreen(screen,board,p1,p2,N1image,N2image,messaget,p1gstats,p1cstats,p2gstats,p2cstats)
        gf.gameover(screen)

        #Display the first turn for 3 seconds

        while pygame.time.get_ticks() - time_clock < 3000:
            screen.blit(messageft, (370, 85))
            pygame.display.flip()
```

Picture 1.5: Rungame functions (main.py)

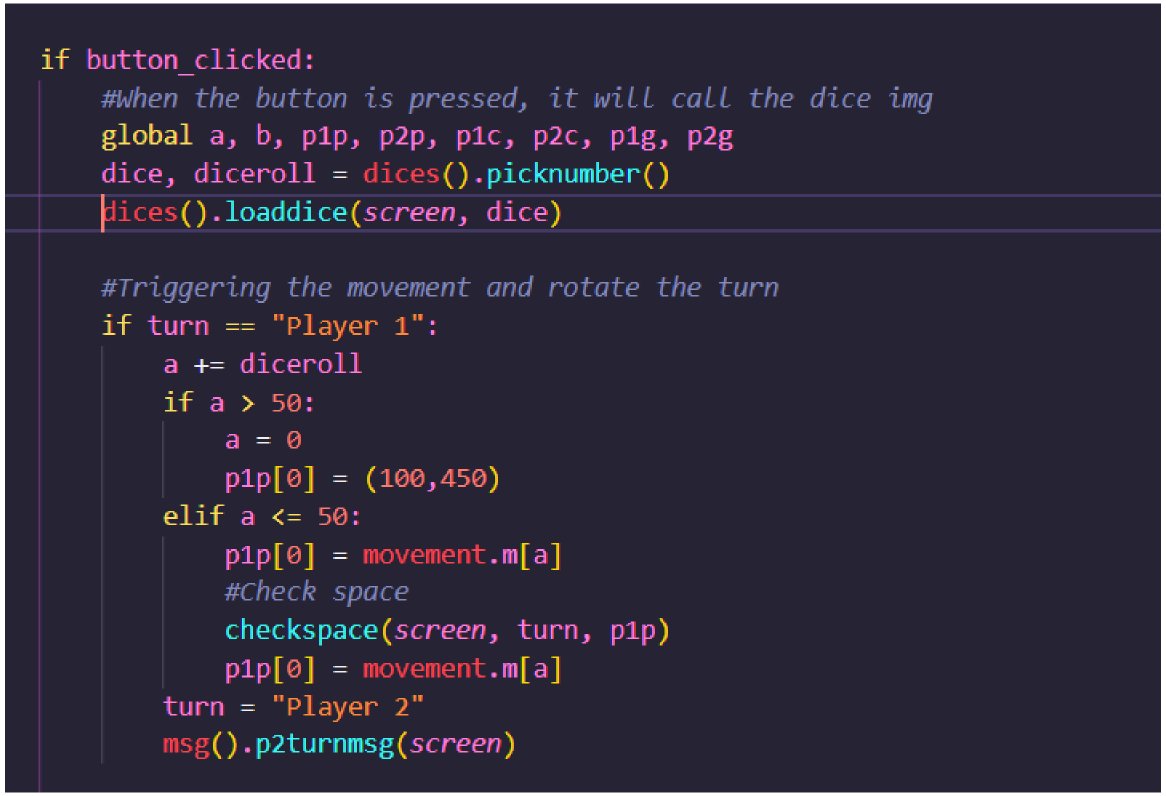
In here, we can see that the program calls the stats every ticks and also check if there is any inputs and also updating the screen such as the player 1 and player 2 image and the stats. Before the game starts, the program will display who has the first turn in the first three seconds. After that, the program also checks that if there is any events occurred. Furthermore, the program will also keep checking if the gameover functions meet the requirement or not. If it is, the game will stop and displays the winner of the game.

In the checkevents function, there are many functions inside of it. Thus, I divide each part in the list.

- 

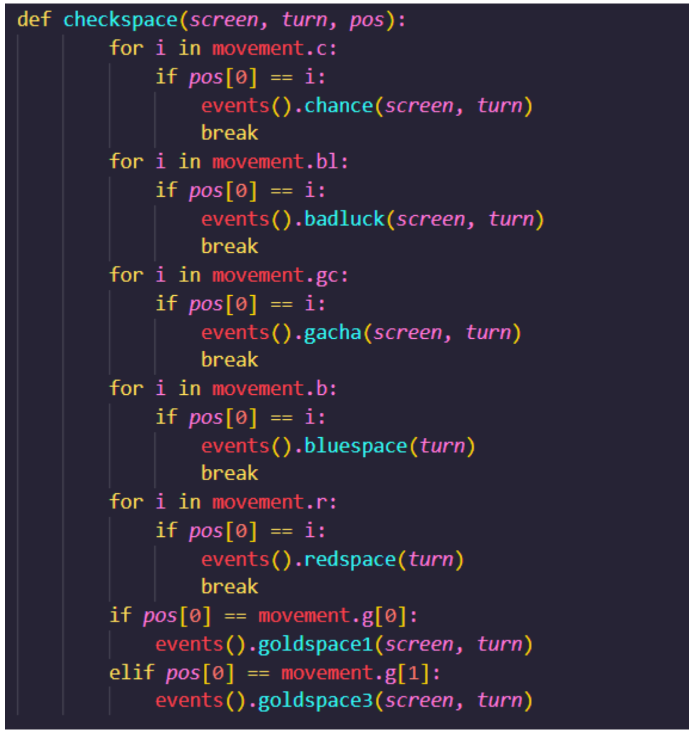
```python
def checkevents(screen, N1pos, N2pos):
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            sys.exit()
        elif event.type == pygame.KEYDOWN:
            check_keydown_events(event)
        elif event.type == pygame.MOUSEBUTTONDOWN and event.button == 1:
            mouse_pos = pygame.mouse.get_pos()
            global turn, red_button_clicked, blue_button_clicked
            if turn == "Player 1":
                button_clicked = N1pos.collidepoint(mouse_pos)
            elif turn == "Player 2":
                button_clicked = N2pos.collidepoint(mouse_pos)
```

In here we can see that, the checkevents function is getting a specific input which is left-click mouse button. If the turn is Player 1, the user can only click the red button next and it is also apply for Player 2, which means they can only click the blue button when it is their turn.

-


```python
if button_clicked:
    #When the button is pressed, it will call the dice img
    global a, b, p1p, p2p, p1c, p2c, p1g, p2g
    dice, diceroll = dices().picknumber()
    dices().loaddice(screen, dice)

    #Triggering the movement and rotate the turn
    if turn == "Player 1":
        a += diceroll
        if a > 50:
            a = 0
            p1p[0] = (100,450)
        elif a <= 50:
            p1p[0] = movement.m[a]
            #Check space
            checkspace(screen, turn, p1p)
            p1p[0] = movement.m[a]
        turn = "Player 2"
        msg().p2turnmsg(screen)
```

See the picture above, when the button is pressed, it will pick a number from 1 to 6 and display the results. Then, it will move the player to the specific space that is determined from the number. Furthermore, it will check the landed space (will be discussed in the next point). Lastly, the player turn will be rotated and displays that the turn has been rotated.

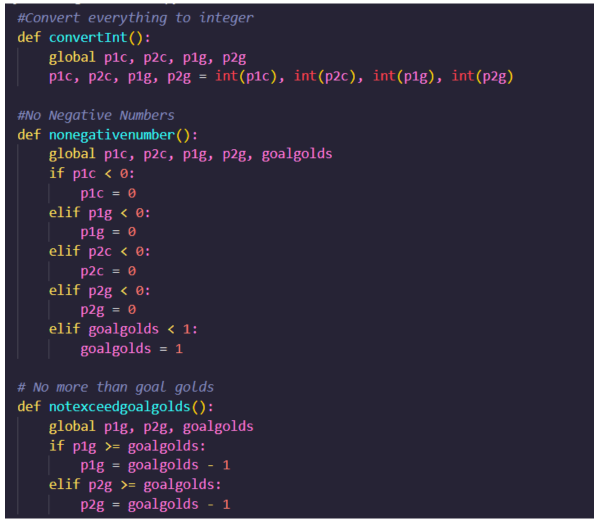
- 

```
def checkspace(screen, turn, pos):
        for i in movement.c:
            if pos[0] == i:
                events().chance(screen, turn)
                break
        for i in movement.bl:
            if pos[0] == i:
                events().badluck(screen, turn)
                break
        for i in movement.gc:
            if pos[0] == i:
                events().gacha(screen, turn)
                break
        for i in movement.b:
            if pos[0] == i:
                events().bluespace(turn)
                break
        for i in movement.r:
            if pos[0] == i:
                events().redspace(turn)
                break
        if pos[0] == movement.g[0]:
            events().goldspace1(screen, turn)
        elif pos[0] == movement.g[1]:
            events().goldspace3(screen, turn)
```

Checkspace function is the one of the main core for the program. When the user lands on chance space, it will trigger a random chance card, if it is a badluck space, the badluck card will appear, and same for the gacha cards. Furthermore, when they land blue space, they will guaranteed get 5 coins and when they land red space they will lose 5 coins too.

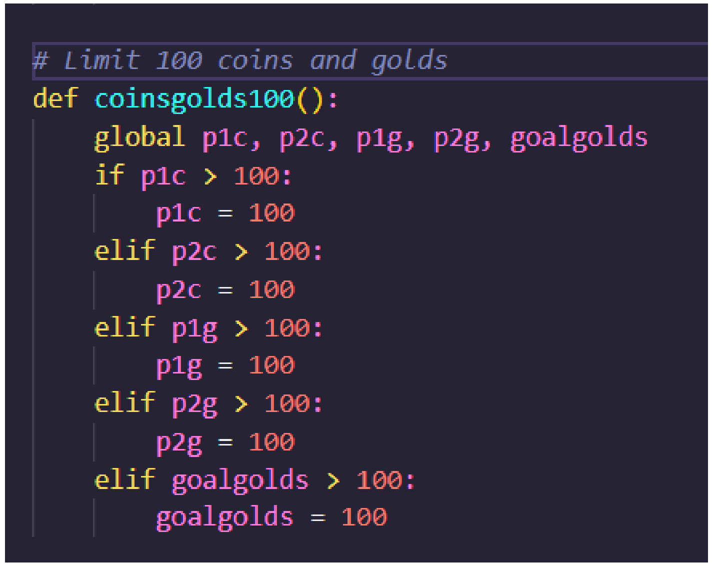
C. Extra Functions

To avoid programs collision, these are the extra functions that I used to make the programs have lesser errors.

- 

```python
#Convert everything to integer
def convertInt():
    global p1c, p2c, p1g, p2g
    p1c, p2c, p1g, p2g = int(p1c), int(p2c), int(p1g), int(p2g)

#No Negative Numbers
def nonegativenumber():
    global p1c, p2c, p1g, p2g, goalgolds
    if p1c < 0:
        p1c = 0
    elif p1g < 0:
        p1g = 0
    elif p2c < 0:
        p2c = 0
    elif p2g < 0:
        p2g = 0
    elif goalgolds < 1:
        goalgolds = 1

# No more than goal golds
def notexceedgoalgolds():
    global p1g, p2g, goalgolds
    if p1g >= goalgolds:
        p1g = goalgolds - 1
    elif p2g >= goalgolds:
        p2g = goalgolds - 1
```

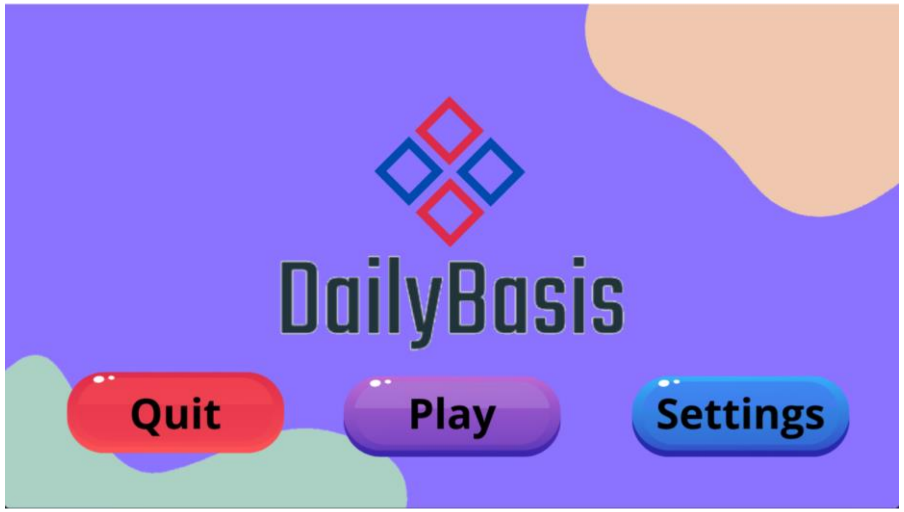
Convertint is used for to convert every float or string to integer. Since we do not want the value of coins and golds are in decimals, we use this function. Furthermore, there is no debt systems in here. Thus, nonegativenumbers come to help which disallow any negative numbers for the coin and gold values. Moreover, we use notexceedgoalgolds function where the handicap golds should not be higher than the goal golds value. If the program does not use this function, the program will trigger the gameover screen instantly.

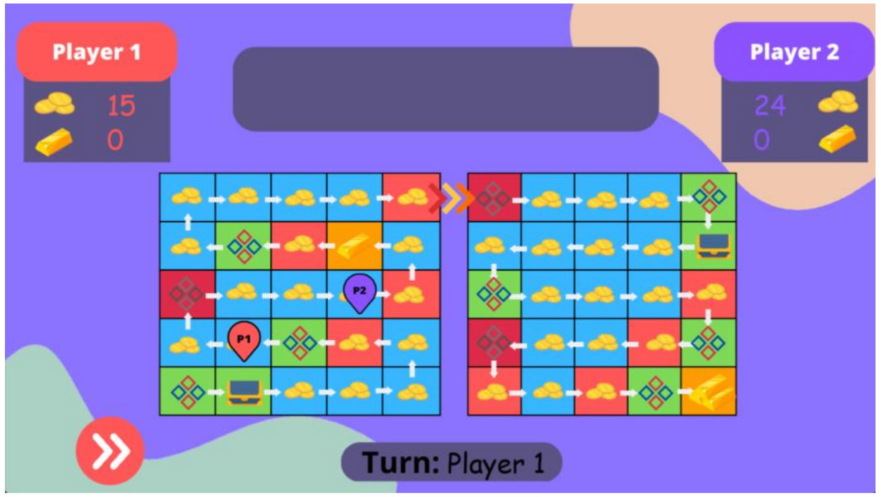
- 

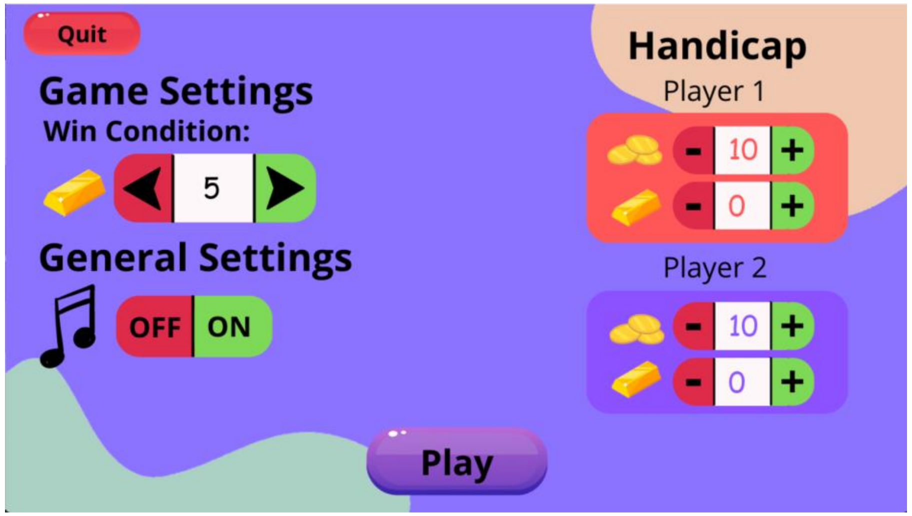
```python
# Limit 100 coins and golds
def coinsgolds100():
    global p1c, p2c, p1g, p2g, goalgolds
    if p1c > 100:
        p1c = 100
    elif p2c > 100:
        p2c = 100
    elif p1g > 100:
        p1g = 100
    elif p2g > 100:
        p2g = 100
    elif goalgolds > 100:
        goalgolds = 100
```

This feature only limits the max handicap of coins and golds are 100 coins and golds. Thus, the texts will not overlap the buttons.

G. Screenshots of the Application



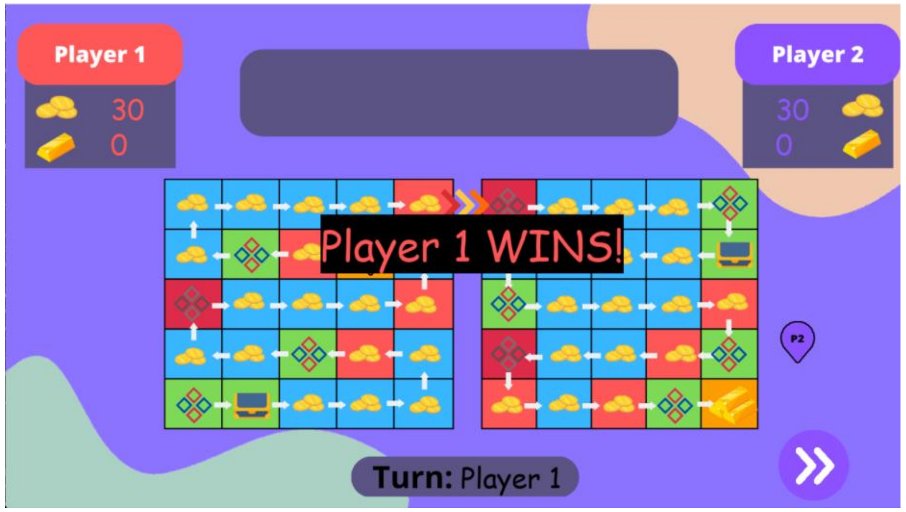




Game Settings
Win Condition: 5

General Settings
OFF ON

Handicap
Player 1: 10, 0
Player 2: 10, 0



H. Lesson learned / Reflection

After doing this huge project, I learned many essential lessons that are useful in the future. In here, I divided two types of lessons, they are lessons that are related to python codes and real life lessons.

A. Python Lessons

- Python is a really fun language and actually easy to understand. By doing this project, it really challenges me to do more advanced and complex for the program. After I built this program, I noticed that classes in python are really helpful in this case. This is due to the fact that not only python classes store values much easier and make the developers knows what they are doing without showing the magic numbers only, but also it can also be used for dividing the functions. Mostly, I used the pygame modules for this program, and most interesting and newest things that I just learned is mixer modules which is a part of the pygame modules. This module can play sound effect or even background music to make the game not deadly silent. Lastly, I also faced a lot of problems when doing if and else, and just learned how to make the multiple conditions in a if statement which I should already have learned before.

B. Real Life Lessons

- Time Management

Some people think that managing time is just a piece of cake, however for me is not. This is because since COVID-19 pandemic, I was really relax and lazy at that time and the homeworks and assigments were not really that many and difficult. Thus, it makes my time management for doing this project is really bad. Doing this project, I learned that time management is really crucial and this applies for other projects such as HCI project and Pancasila project.

- Socials

Many people who has the “quiet” habit including me mostly do their task alone even if it is group project. Although this is a individual project, I found out that communicating and discussing about this project is a good idea to improve my project. When I was communicating with my deskmates and friends about their projects, I saw their projects and the source codes which really helps me to solve the long-term errors.

I. Program/Source Code

https://github.com/DNeilson67/DailyBasis

J. 3-5 Minutes video demo link

The video is available in Github:
https://github.com/DNeilson67/DailyBasis