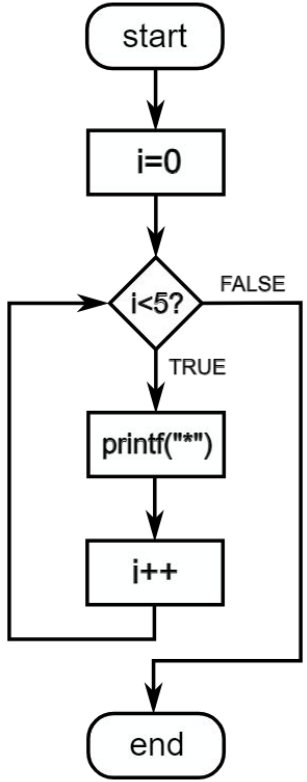
Programming Fundamentals

With Python

Chapter 4

01

For Loops



Basic Structure

```
1 for index in range(0,5):
2     print(index)
```

- index is a local variable, which means it is a variable that you can only access within the for loop.
- range accepts 2 arguments: first one (0) is the start, and second one (5) is the end.
- The for loop will iterate from start (0) upto but not including the end (5). So, it will stop when index = 4.

Output

```
0
1
2
3
4
```

Basic Structure

```
1 for index in range(5):
2     print(index)
```

If you give one argument, python will assume that you want the start to be 0 by default.

Output

```
0
1
2
3
4
```

Basic Structure

You can add a third argument that describes how much you will increment by. If it is a positive number, index will increase, and if it is a negative number, index will decrease.

```
1 for index in range(0,10,2):
2       print(index)
```

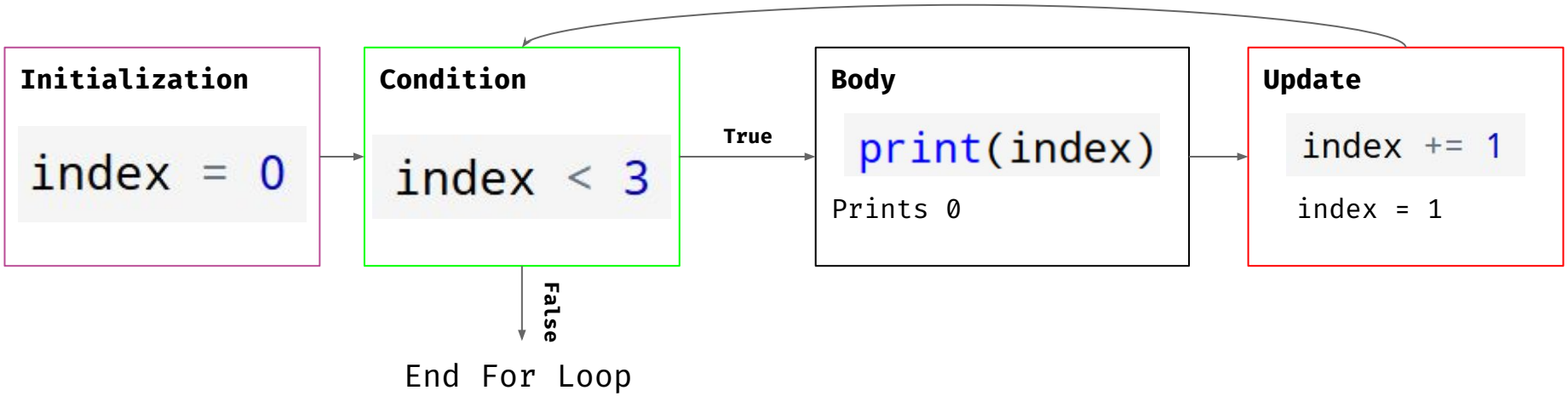
```
0
2
4
6
8
```

```
1 for index in range(3,0,-1):
2       print(index)
```

```
3
2
1
```

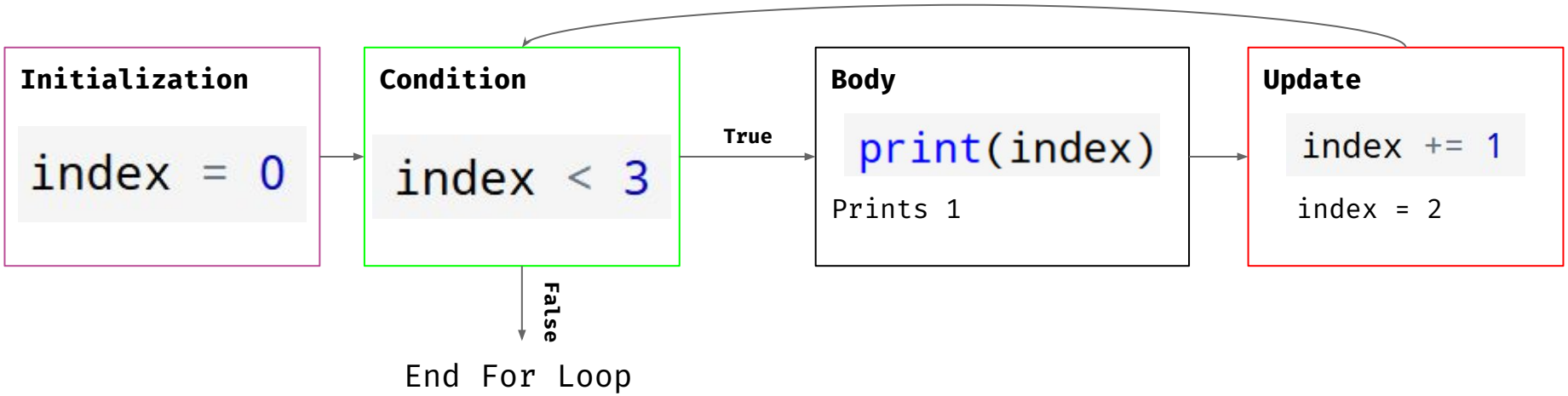
How Does The For Loop Work?

```
1 for index in range(0,3):
2     print(index)
```



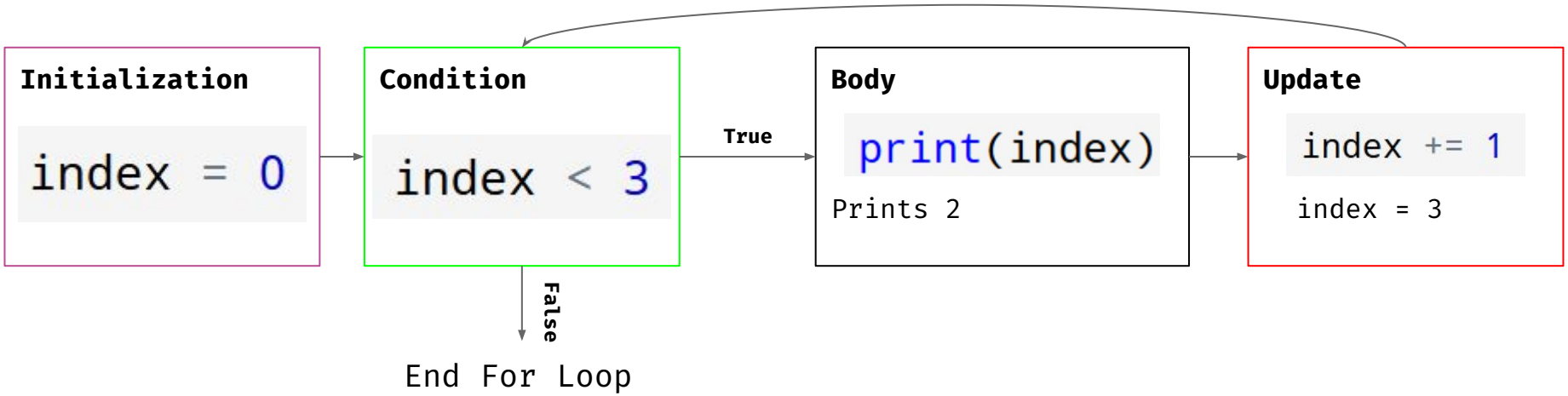
How Does The For Loop Work?

```
1 for index in range(0,3):
2     print(index)
```



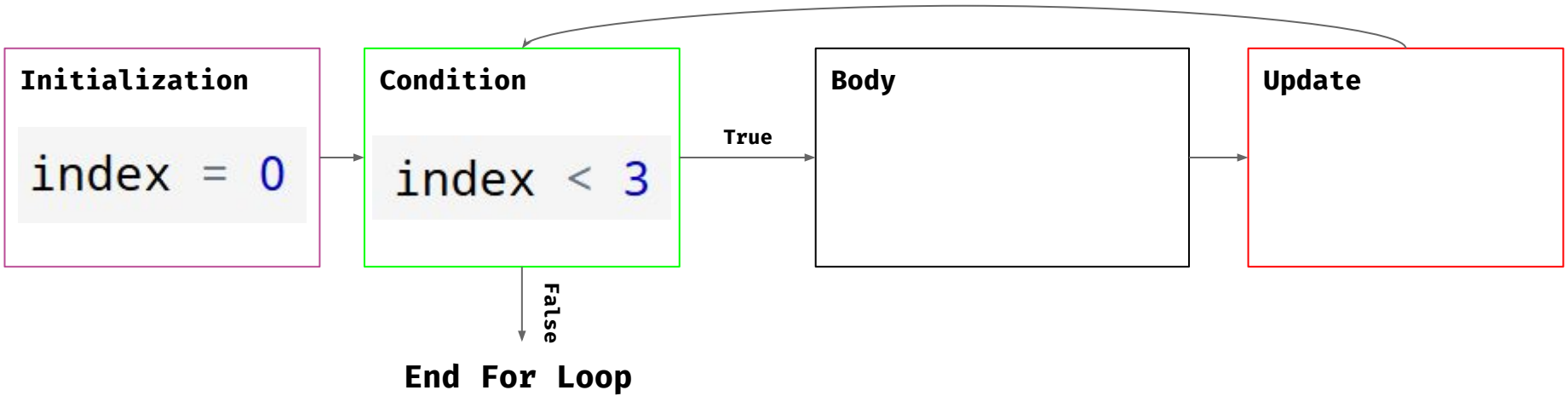
How Does The For Loop Work?

```
1 for index in range(0,3):
2     print(index)
```



How Does The For Loop Work?

```
1 for index in range(0,3):
2     print(index)
```



How Many Times will the code print “Hello”?

```
1 for i in range(2,10,2):
2     print("Hello")
```

4

```
1 for i in range(3,0):
2     print("Hello")
```

0 - i initializes to 3, then it checks if 3 < 0, then stops iterating. So, it never goes to the body.

```
1 for i in range(99,237):
2     print("Hello")
```

138

```
1 for i in range(11,2,-1):
2     print("Hello")
```

9

```
1 for i in range(11,2,-3):
2     print("Hello")
```

3

For Loop Exercise 1

A new calculator model is coming out and they want somebody to implement the permutations formula in their calculator. The formula for permutations is

nPr = $\frac{n!}{(n-r)!}$ where r is the number of objects to select from n, which is total number of objects. The permutation calculates number of different ways to choose from n objects r amount each time.

Guidelines:

- Use user input to get value of n and r that user wants.
- Permutations cannot be calculated if n < r, so make sure n ⩾ r. Print an error statement if n < r and the result if n ⩾ r.
- There will never be a need to use double or float in this program.
- Use of functions is a must, as you will be implementing similar logic more than once.
- Make sure result is not too big, as factorial can result in a huge number quite fast, and there is limited memory for a variable.

Example Input and Output:

```
Please enter value of r: 6
Please enter value of n: 12
The result is 665280
```

```python
def calculateFactorial(factorialMax):
    result = 1

    for i in range(1, factorialMax + 1):
        result *= i

    return result

n = int(input("Enter value of n: "))
r = int(input("Enter value of r: "))

if n >= r:
    numerator = calculateFactorial(n)
    denominator = calculateFactorial(n - r)

    print(f"The result is {numerator/denominator}")
else:
    print("The variable n must be greater than or equal to
        r")
```

How Nested For Loops Work?

You can think of nested for loops as outer loop is the row # and inner loop is the column #. You can also think of it as outer loop is a list of lists.

What is the output?

```
1 for i in range(1,4):
2     for j in range(1,4):
3         print(f"{i*j} ", end = '')
4     print('')
```

```
1 2 3
2 4 6
3 6 9
```

How Nested For Loops Work?

You can think of nested for loops as outer loop is the row # and inner loop is the column #. You can also think of it as a matrix to help you follow the code.

```
1 for i in range(1,4):
2     for j in range(1,4):
3         print(f"{i*j} ", end = '')
4     print('')
```

i = 1 j = 1 i * j = 1	i = 1 j = 2 i * j = 2	i = 1 j = 3 i * j = 3
i = 2 j = 1 i * j = 2	i = 2 j = 2 i * j = 4	i = 2 j = 3 i * j = 6
i = 3 j = 1 i * j = 3	i = 3 j = 2 i * j = 6	i = 3 j = 3 i * j = 9

Nested For Loops - How Many Iterations or what is the value of counter at the end?

```
1  counter = 0
2
3  for i in range(2,50):
4      for j in range(50,61):
5          counter += 1
6
7  print(counter)
```

(50 - 2)*(61 - 50) = **528**

For Loop Exercise 2

Your task is to build a mario ladder with hashtags. The way it will work is:

1. Get user input for the height of the mario ladder they want
2. Draw the mario ladder with the appropriate height using hashtags.

Example

```
Enter the height of the mario ladder: 4
#
##
###
####
```

```
height = int(input("Enter the height of the mario ladder: "))

for i in range(height):
    for j in range(i + 1):
        print("#", end = '')
    print('')
```

For Loop Exercise 2

Your task is to build a mario ladder with hashtags. The way it will work is:

1. Get user input for the height of the mario ladder they want
2. Draw the mario ladder with the appropriate height using hashtags.

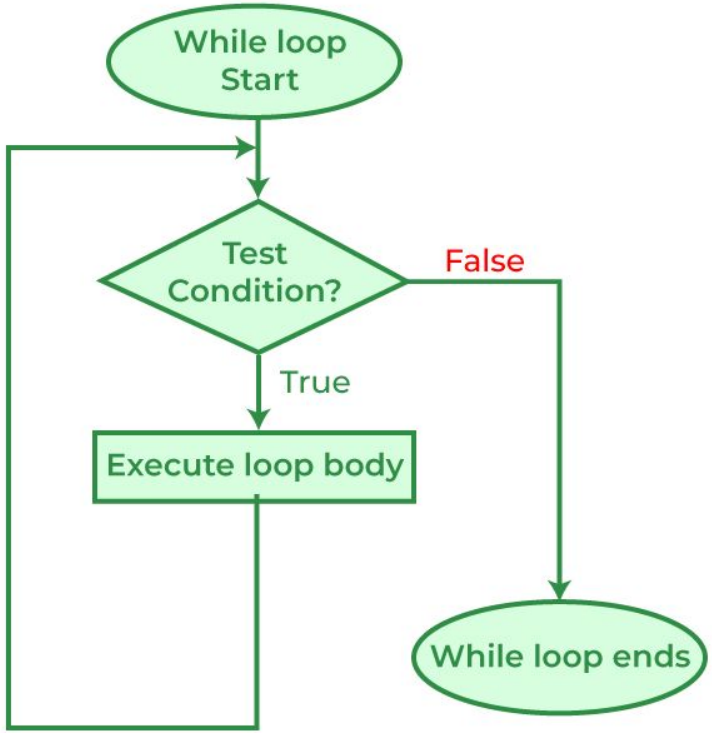
Example

```
Enter the height of the mario ladder: 4
#
##
###
####
```

i = 0 j = 0 **#**			
i = 1 j = 0 **#**	i = 1 j = 1 **#**		
i = 2 j = 0 **#**	i = 2 j = 1 **#**	i = 2 j = 2 **#**	
i = 3 j = 0 **#**	i = 3 j = 1 **#**	i = 3 j = 2 **#**	i = 3 j = 3 **#**

02

While Loops



Basic Structure

```
1  i = 0
2
3  while (i < 3):
4      print("Hello")
5      i += 1
```

Output

```
Hello
Hello
Hello
```

Condition: This is where you write a boolean expression. If it outputs True, it goes into the body, otherwise, it stops looping.

NOTE: Always make sure the variable being checked in the while loop condition will always be updated at some point inside the body of the while loop and will make the condition false at some point. Otherwise, it will loop forever!

When to use While Loop vs. For Loop?

For Loop	While Loop
When you know ahead of time how many times to loop.	When you don't know how many times you'll need to loop. So, the condition depends on an update inside the loop.
	If you try using a for loop and you want to put an if statement that breaks the loop
	When you want to check if a certain variable reaches a certain value but you do not know when it will reach that value (e.g., user input, searching through something, etc.)

While Loop Example

Write a program that keeps looping until you want to exit like the output seen on the right.

Output

```
Enter 1 to exit and 0 to continue: 0
Enter 1 to exit and 0 to continue: 0
Enter 1 to exit and 0 to continue: 0
Enter 1 to exit and 0 to continue: 1
You have successfully exited!
```

```
1  exit = False
2
3  while not exit:
4      exit = int(input("Enter 1 to exit and 0 to continue: "))
5
6  print("You have successfully exited!")
```

While Loop Example

```
1  i = 9
2
3  while i > 0:
4      print("Hello")
5      i /= 3
```

Output

It will infinitely print “Hello” because 9/3 = 3 then 3/3 = 1 then it keeps dividing by 3 and gets closer to 0, but it never reaches 0.

While Loop Example

```
1  j = 3
2  i = 1
3
4  while i + j == 4:
5      print(i, j)
6      if (j > 0):
7          j -= 1;
8      i += 1
```

Output

```
1 3
2 2
3 1
4 0
```

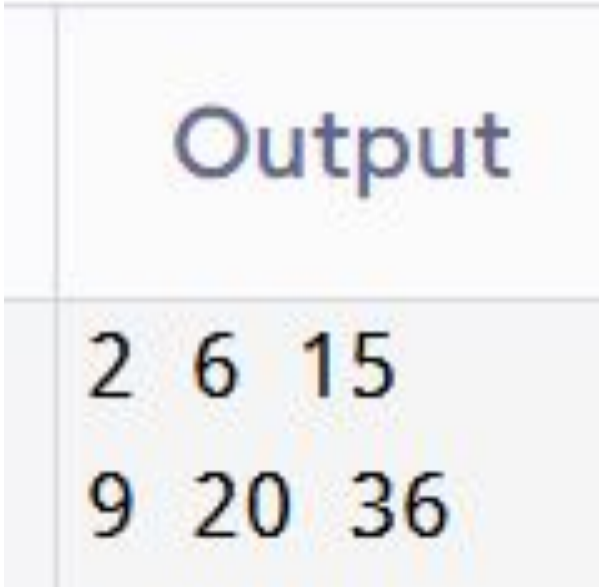
While Loop Example

```
1  j = 3
2  i = 1
3
4  while i + j == 4:
5      print(i, j)
6      if (j > 0):
7          j -= 1;
8      i += 1
```

i = 1	j = 3	i + j = 4	Initialization First Condition Check
i = 2	j = 2	i + j = 4	Second Condition Check
i = 3	j = 1	i + j = 4	Third Condition Check
i = 4	j = 0	i + j = 4	Fourth Condition Check
i = 5	j = 0	i + j != 4	Fifth Condition Check. Since condition not met, this iteration doesn't run

While Loop Example

```
1  counter = 0
2  i = 2
3
4  while i < 10:
5      j = 1
6      while j <= 3:
7          print(i*j,end = ' ')
8          i += j
9          j += 1
10     i += 1
11     print('')
```



While Loop Example

```
1  counter = 0
2  i = 2
3
4  while i < 10:
5      j = 1
6      while j <= 3:
7          print(i*j,end = ' ')
8          i += j
9          j += 1
10     i += 1
11     print('')
```

i = 2 then i = 3 j = 1 i* j = 2	i = 3 then i = 5 j = 2 i* j = 6	i = 5 then i = 8 j = 3 i* j = 15
i = 9 then i = 10 j = 1 i* j = 9	i = 10 then i = 12 j = 2 i* j = 20	i = 12 then i = 15 j = 3 i* j = 36
i = 16 Then condition is not met, so it stops here		

Loop Exercise

You are going to implement the classic guessing game. It is gonna test your coding skills from the beginning of the course up until now.

Guidelines

- It is up to your judgement when to use if statements, functions, for loops or while loops. Remember that there isn't an objectively right answer, but I will give you some suggestions.
- Make a main function for the game menu, where user can pick an option and based on that option, user can do something. I recommend using a while loop for this.

```
/* Option 1 => Play the Guessing Game
   Option 2 => Show game stats. So,
   number of games played, number of wins, and number of losses.
   Option 3 => Exit the game.
*/
```

Loop Exercise

You are going to implement the classic guessing game. It is gonna test your coding skills from the beginning of the course up until now.

Guidelines

- I recommend using if statements for checking which option the user picked.
- I recommend separating the actual game in a function called play(), where you continuously ask user to guess the number for a certain number of attempts. If user guesses too high or too low or guesses correct, make sure to tell them. This logic should be put in a while loop. Also the play function should return a boolean, where if it returns a 0 or False, the player lost the game, and if it returns a 1 or True, the player won the game.

Loop Exercise

This is how you can implement a computer picking a random number:

```
1  import random
2
3  number = random.randint(1,10)
4
5  print(f"The computer picked: {number}")
```

Loop Exercise

This is how you can make a main function:

```
1 def main():
2     print("This is the main function")
3
4 if __name__ == "__main__":
5     main()
```

Loop Exercise

Example of how the game should look like:

```
-----------------------------------
Option 1 => Play the Guessing Game
Option 2 => Show game stats
Option 3 => Exit the game
-----------------------------------
Enter an option from 1 to 3: 1
You have 4 tries
Guess a number from 1 to 10: 5
Your guess was too low.
You have 3 tries
Guess a number from 1 to 10: 7
Your guess was too low.
You have 2 tries
Guess a number from 1 to 10: 8
Your guess was too low.
You have 1 tries
Guess a number from 1 to 10: 9
Your guess was too low.
You lost. The number was 10.
```

```
-----------------------------------
Option 1 => Play the Guessing Game
Option 2 => Show game stats
Option 3 => Exit the game
-----------------------------------
Enter an option from 1 to 3: 1
You have 4 tries
Guess a number from 1 to 10: 5
Your guess was too high.
You have 3 tries
Guess a number from 1 to 10: 3
Your guess was too high.
You have 2 tries
Guess a number from 1 to 10: 2
Your guess was too high.
You have 1 tries
Guess a number from 1 to 10: 1
Congrats! You guessed the number correct
```

```
-----------------------------------
Option 1 => Play the Guessing Game
Option 2 => Show game stats
Option 3 => Exit the game
-----------------------------------
Enter an option from 1 to 3: 2
Number of Games: 2 Number of Wins: 1 Number of Losses: 1
-----------------------------------
Option 1 => Play the Guessing Game
Option 2 => Show game stats
Option 3 => Exit the game
-----------------------------------
Enter an option from 1 to 3: 3
Thank you for playing the guessing game!
```