50 Programming Assignments

Practice your coding

HARMAN ABDUL WAHEED

Assignment

Instagram: fly_fazaia

harmanwaheed@gmail.com

Welcome message

Good luck starting this course! Remember your purpose is not to just solve assignments but to understand these. Some assignments will require more time and some will not. Be consistence. At least do 2-3 assignments per day. Good luck with this journey!

Description

Hi there, in this project of 50 assignments you will get to **solve assignments related to simple python, conditional statements, loop structures, functions and at the end there are assignments in which we have integrated all topics.** This course is specially designed for python but one can solve the assignments in other programming language as well.

In case you won't be able solve the assignment you can navigate to the **solution segment to find the solution of the assignment.** We have managed all the assignments and there solutions in a well managed way. You can contact us to access the word document of all the assignments and solution of all assignments is already available in shared github repository.

Good luck

Distribution of assignments:

Assignment 1 – Assignment 5 -> **Level 1, Basic python**

Assignment 6 – Assignment 12 -> **Level 2, Conditional statements**

Assignment 13 – Assignment 21 -> **Level 3, For loops**

Assignment 22 – Assignment 26 -> **Level 4, While loops**

Assignment 27 – Assignment 40-> **Level 5, Functions**

Assignments 41 – Assignment 50 -> **Level 6, everything integrated**

Take your time and understand all the assignments. Once you are done with each assignment of this course, you will hopefully be able to start your professional journey or journey of data science toolbox. Furthermore you will be able to make professional menu tasks. If you are willing for that, I can provide you a task of school management system task which will be very wide and consist of many options. We made that project on the requirement of Malaysian client. We can also provide you with code of that project.

SECTION 1: Python Basic

Instructions

This section will contain python basic assignments.

Objectives of this particular section:

- becoming familiar with the concept of storing and working with different data types in Python;
- Experimenting with Python code.
- Becoming familiar with the concept of variables.
- Performing basic math and conversions.

1 – Simple Variable

Once upon a time in England, John had 10 coins, Mary had five, and Adam had six. They were all very happy and lived for a long time. End of story.

Your task is to:

- create the variables: john, mary, and adam;
- Assign values to the variables. The values must be equal to the numbers of coins possessed by John, Mary, and Adam respectively;
- Having stored the numbers in the variables, print the variables on one line.
- Now create a new variable named total_coins equal to addition of the all the coins.
- Print the value stored in total_coins.

2 – Miles/Kilometers conversion

Miles and kilometers are units of length or distance.

Bearing in mind that 1 mile is equal to approximately 1.61 kilometers, **complete the program in the editor so that it converts:**

- **Miles to kilometers;**
- **Kilometers to miles.**

You have to complete this program:

```
kilometers = 12.25
miles = 7.38

miles_to_kilometers = ###
kilometers_to_miles = ###

print(miles, "miles is", round(miles_to_kilometers, 2), "kilometers")
print(kilometers, "kilometers is", round(kilometers_to_miles, 2), "miles")
```

Do not change anything in the existing code. Write your code in the places indicated by ###.

In upper code, we created two variables on which conversion is applied later on.

After completing the code, run the program and see what happens.

---

3 – Calculator

input a float value for variable a here

input a float value for variable b here

output the result of addition here

output the result of subtraction here

output the result of multiplication here

output the result of division here

print("\nThat's all, users!")

Your task is to complete the code in order to evaluate the results of four basic arithmetic operations.

Write your code in place of # according to the given instructions.
Note that \n in the print statement in next line character. It will skip one line and output 'That's all, users!' will be printed on next line

---

4 – Temperature Unit Converter

- Write a Python program that converts temperature from Celsius to Fahrenheit. Your program should:

- Prompt the user to input a temperature in Celsius. (take input)

- Calculate the equivalent temperature in Fahrenheit using the formula: $F = 9/5 \times C + 32$.

- Print the result.

Don't hesitate to use as many brackets in calculation as you want.

---

5 – BMI calculator

Doctor Ahmad needs a program in which patient enter his weight and height, and program should return BMI value of that patient.

Your strategy should be:

- Ask the user to input their weight in kilograms.

- Ask the user to input their height in meters.

- Calculate the BMI using the formula: $BMI = weight / height**2$. (height is squared in formula)

- Output the BMI value to the console.

SECTION 2: Conditional Statements

6 – Conditional statements

Write a Python program that prompts the user to input a score.

Program should prints out "you are passed" if the user has scored more marks than 40.

These are known as **if statements.**

ARE YOU DONE WITH THIS?

Now, update you program, it should be able to deal with other scenario as well. If the user enter numbers less than 40, the program should print out: "You are failed, better luck next time."

These type of conditional statements are known are **if else statements.**

Now gear up for the coming assignments.

---

7 – Grading System

Write a Python program that prompts the user to input a score and then prints out the corresponding grade according to the following criteria:

- If score is greater than 90, grade is A
- If score is between 80 and 90, grade is B

- If score is between 70 and 80, grade is C
- If score is between 60 and 80, grade is D
- If score is less is than 60, grade is E

- If score is less than 0 or greater than 100, Program should tell the user that **"marks are out of range. You score should be in between 0 and 100."**

Such statements are known as **if else if** or **else if statements.**

8 – Find largest number

Write a Python program that asks the user to input three numbers and then prints out the largest among them.

9 – Swapping values

Write a program that ask user to input two numbers and store the values in **x** and **y.**

Your program should then swap the values such that x will get value of y and y will get value of x.

Unfortunately you can't create a third variable for this task.

sample:

enter a number: 5   (store value in x)
enter another number: 6   (store value in y)
Values swapped!

x = 6
y = 5

This task can take your time. For better understanding take out your notebook and imagine different scenarios. Furthermore, this task is more related to the mathematical concepts.

Give time to this assignment and properly understand your code.

___

10 – Triangle type checker

Create a Python program that determines the type of triangle based on its sides. Your program should:

- Prompt the user to input the lengths of three sides of a triangle.
- Use conditional statements to check if the triangle is equilateral, isosceles, or scalene.
- Print the type of triangle.

Equilateral triangle: Three sides are of same length.

Isosceles triangle: Two sides are of same length.

Scalene triangle: All three sides are of different lengths.

You can use as many conditional statements as you want.

___

11 – Tax Calculator

In a City name *Jabinga* in *molangala* have heavy taxes on monthly income of each of there employee.

There rules are:

- 0% tax on income up to $300
- 10% tax on income from $400 to $700
- 20% tax on income from $700 to $1000
- 30% tax on income above $1000

Ten thousand of the total population was assigned just to calculate tax of each individual. Still sometimes few people used to claim that the calculations are wrong.

You need to create a program that takes income as input from user and display the amount tax that this particular person has to pay.

Let me break down steps for you:

1- User input

2- Tax calculations

3- Output

It is something interesting. Good luck.

Note: no such states exist in the world.

12 – Factors

As a math teacher, you need to write a program for your students in which your student will input a number and program will indicate whether it is a multiple of 2, 3, 4, 5, 6, 7, 8, 9 and 10. In short it will give all factors of that number from 1 to 10.

Students can give any number. For that number your program should return all factors from 1 to 10.

For example if the user enters 20, our program should print all numbers that are factor of 20. In this case it will print: 1, 2, 4, 5, 10.

If the user enter 15, our program should print: 1, 3 and 5.

SECTION 3: For loops

13 – Introducing for loops

1. Write a python program that prints the number from 1 to 100 using for loop
2. Write a python program that prints the sentence "Your name is professional in python" 100 times. Replace "Your Name" with your actual name

These will be two different programs.

14 – Loops and lists – handling students data

Scenario: A school class contains 20 students. Write a Python program using a `for` loop that prompts the user to enter the name of a student 20 times. Each entered name should be stored in a list named `students`.

Requirements:

- Initialize an empty list named `students` before the loop starts.
- Print the list after each iteration to show how elements are being added dynamically.

Hint: Adding elements in a list: *list.append(element)*

---

15 – Calculating Average of whole class

Let's calculate average marks of a class using loops.

Scenario: Let say class consist of 20 students. For loop should run 20 times and ask user to enter marks of student each time. Store all marks in a list.

Implementation: Once data of 20 students in entered, you have list containing marks of all 20 students. Calculate average of class by this formula: *Sum of numbers / Total number of student*

Length of list will represent total number of students

You can use built function sum(list) and len(list) while calculating the average value.

---

16 – For loops and Conditional statements - Merged

Let's merge `conditional statements` and `for loops`. Your task is to write a program that execute `for loop` 10 times. Each time it should:

- Take number a input from user
- Apply if else block to check whether number is an even number or odd number
- If it is an even number, print *'Number is even!'* and if number is odd, print *'Number is odd!'*

---

17 – Finding sum of list manually

li = [34, 54, 23, 05, 45, 65]

Your task is to write a `for loop` that calculate sum of all numbers of the given list.

With every iteration of loop, print the current sum to see what's happening…

---

18 – Multiplication table

Multiplication table

Your program should take number from user and output table of that number up to 10.

If input number is 3, output should look like:

3 X 1 = 3

3 X 2 = 6

3 X 3 = 9

3 X 4 = 12

3 X 5 = 15

3 X 6 = 18

3 X 7 = 21

3 X 8 = 24

3 X 9 = 27

3 X 10 = 30

---

19 – Handling students data in lists

Student's info

Create two empty lists: *student_name* and *student_marks.*

Write `for loop` in which user will give name and marks of students as input and you will store name in *student_name* list and marks in *students_marks* list.

Enter data for 10 students.

Write another `for loop`. This `for loop` will print each student's name with their marks in a logical way.

Sample output will be:

Ali got 45 marks

Ahmad got 60 marks

Mujtaba got 46 marks

Rehman got 60 marks

Murtaza got 34 marks

Abdullah got 40 marks

Hufaiza got 90 marks

Muneeb got 50 marks

Fahad got 56 marks

Zain got 79 marks

Names and marks of students are changeable. You will give these as inputs during execution of first `for loop.`

---

20 – Finding Factorial

In grade 11, we had to find factorial of a number.

I was fun for smaller number, but for larger numbers it was lengthy and boring and sometimes calculations were wrong due to human error. **Let's now create a program that will calculate factorial of a given number.**

What is factorial?

Factorial of 3 = 3 X 2 X 1 = 6

Factorial of 4 = 4 X 3 X 2 X 1 = 24

Similarly any number multiplied by all numbers smaller than that number gives factorial value.

For example if number is 5, then 5 will get multiplied by all numbers smaller than 5 (including 5) i.e. 1,2,3,4, 5

Factorial of 5 = 5 X 4 X 3 X 2 X 1 = 120

Write a python code that will just take number as input and print its output.

Let's make this program together!

```
product = 1

# initially factorial is 1

fact = ____('Enter a number you want factorial of: '))

# Take input from user in integers.

# Initialize for loop. Range of i should cover all number less than input but
it should not include 0

# because 0 is not included in calculation of factorial. if input is 3, range
should be: 1, 2, 3. Range

# should include input itself as well.

for i in range(_, fact+_):

# with every iteration, multiply value of product with current value of i and
rewrite value of product
```

```
    product = product * __

print('Factorial of given number is: ' + str(product))
```

- Complete the code according to the mentioned instructions in comments.
- After completing the code, you can remove the comments (as per your feasibility).

21 – Break Statement in loops

Emergency - Break statement

Break (break) keyword in loops is used to stop the execution. Now we will write a program that will run 100 times. But under certain conditions its execution will stop immediately.

Scenario: Let say you are a doctor. Whenever a patient comes, one of your worker enter disease name of that particular patient.

You want your system to stop taking input once the Heart patient comes. Whenever the disease name is **'heart'** then the program should print **'Emergency situation Emergency situation'** and no more inputs will be taken.

Strategy: Write `for loop` that will run 100 times. Under the body of loop, take disease name as input from user. Use if statement to check whether the disease name is **'heart'** or not, if so, then handle the condition accordingly.

SECTION 4: While loops

22 – Introducing while loops

While loops

Let's start `while loops` in python. We will start from making simple while loop,

- Write a while loop that print numbers from 1 to 20.
- Write a while that print multiples of 3 from 30 to 90. (3, 6, 9, … 90)
- Write a while loop that print your name one hundred times.

Yes, these are 3 separate tasks.

Output for third program should look like:

1 - Ali

2 - Ali

3 - Ali

.

.

.

.

100 – Ali

Ali will be replaced with your name.

---

23 – Atomic Power – Guess game

Guess game – atomic power

Let's create a simple guess game using while loop.

Every guess game have a particular question and its answer. Our question will be:

'Which state did first atomic attack on other state?' and answer to this question is: *America*.

Write a while loop that will ask user the question *'Which state did first atomic attack on other state?'* Again and again until the user enters 'America. Once we get the right answer, we will get out of while loop, because the question is answered.

Within the body of while loop, take input from user and store the value in a variable. There are multiple ways to write such code. Try building your own logic.

Sample output:

```
Which state did first atomic attack on other state? Britain
Which state did first atomic attack on other state? Egypt
Which state did first atomic attack on other state? Iran
Which state did first atomic attack on other state? Japan
Which state did first atomic attack on other state? China
Which state did first atomic attack on other state? America
```

]:

24 – Designing a simple quiz

Professor Amjad need a program for his math students. Students are aged 7-13. Professor wants
to make a program that will ask the student simple multiplication questions and keep asking the
question until correct answer is entered. To generate two random numbers, we will use random
module.

```
import random
a = random.randint(1,10)
```

These lines will generate a random number from 1 to 10.

Step by step guide

- You will write a program which will generate two random numbers. Store your random
  number in variable *'num1'* and *'num2'*.
- Create a third variable *'product'* equal to product of two random number that we
  generated previously.
- Create variable *'answer'*. This variable will be empty string.
- Finally write a while loop that will run until answer is not equal to *'product'* variable.
  Within the while loop, ask question from the user and store answer in variable *'answer'*.

- Question is: num1 X num2 = ?

(num1 and num2 are randomly generated numbers.)

Sample output:

```
3 X 4 = 11
3 X 4 = 13
3 X 4 = 15
3 X 4 = 54
3 X 4 = 12
```

25 – Designing a simple quiz - continued

Let's make previous assignment more practical. Let say professor want us to make a program that ask the questions, but this time, ask the questions multiple times (with different values each time). Let say he want us to ask 15 questions from student.

How can we implement this? You can copy the code and paste it again and again and hence the program will ask questions again and again.. Right?

Well, try not to do that way! Use a `for loop` to repeat the process 15 times.

Hint: Write a `for loop` that will execute 15 times and place your whole code within body of that for loop.

---

26 – Sales track

There are three seating categories at a stadium.

- Class A seats cost 2000 rupees,
- Class B seats cost 1500 rupees,
- Class C seats cost 1000 rupees,

Write a program that asks how many tickets for each class of seats were sold, then displays the amount of income generated from ticket sales.

You may implement this without using loop structure.

Section 5: Functions

27 – Simple Function – User intro

create a function 'profile' with two parameters: name and age. The function should print:

'Murtaza is 35 years old.' (Murtaza and 35 are arguments that we will pass when calling the function)

```
# complete function by filling the blanks
def profile(--,--):
    print(---)

# Call the function
```

28 – Saving time by using functions

Let suppose you have a program:

```
a = input('Write your first name: ')
b = input('Write your last name: ')

print('\nThanks for your cooperation')
```

Output:
Write your first name: harman
Write your last name: waheed

Thanks for your cooperation

You are making an application and this part of code will repeat again and again. To save the time, and instead of writing these three statements again and again, You can convert this into a function

Define a function 'into' with no parameter. And place the upper code in body of the function. Make any changes if needed and check the program by calling function.

29 – Efficient calculator in a function

Let's now create a function calculate(), which takes two numbers as parameters. These parameters will be 'num1' and 'num2'

This function will print result of addition, subtraction, division and multiplication of those two numbers

```
# complete the code

def calculate(--, --):

    print('Addition of given numbers:', ---)
    print('\nSubtraction of second number from first:', ---)
    print('\nMultiplication:', ---)
    print('\nFirst number divided by second:', ---)

# call your function with values of num1 and num2 as 2 and 2 and analyze the results.
```

30 – Predefined parameters in functions

We have created a function is 27th assignment. Let say all persons are of age 40 in that assignment. Very few are those who are not 40.

Make some changes in the function such that, it will automatically read age as 40 when the program is called.

Also, when in some cases we want to change the age, our program should handle those situations by passing the age as first argument when calling the function

Hint: no hint

31 – Calling functions with keyword and positional arguments

```
def func(a, b, c, d):
   print(a)
   print(b)
   print(c)
   print(d)
```

Call this function in two ways:

1 - give positional arguments

2 - give named (keyword) arguments

When giving named argument, go against the order. For example, supply b first, then d, then c and then a.

---

32 – Debug the function

Debug the code given below.

```
def range(number):
    for i in range(10):
        print()
```

Functionality should be: this function will print all numbers smaller than the given number as a argument.

33 – Practice predefined parameters

When we define a function with parameters, predefined parameter is always placed on end. Write a function 'players' with two predefined parameters and one simple parameter.

-----------**Parameters**-----------

name

suports = 'cricket'

standard = 'great'

```
def ---(---,---,---):
    print(---, 'is a', ---, ---'player')
```

Update the print statement such that output when calling the function should look like:

Pat cummins is a great cricket player.

Note that 'Pat cummins', 'great' and 'cricket' are arguments that we will pass to the function.

---

34 – Calling function practice

After you are done with assignment 33 function, let's call the function you created.

Run your function in 5 times in a way thar each time your output looks like this:

1 - Messi is a worst cricket player

2 - Messi is a astonishing football player

3 - Babar azam is a unpredictable cricket player

4 - Akmal shehzad dogar is a strong kabaddi player

5 - Azam khan is a active cricket player

---

35 – Writing a simple function

Write a function 'motivate' with only one parameter 'iterr'.
By default, value of iterr should be 10

What function will do: it will print 'I can do it' as many times as specified in the argument.

---

36 – Make code reusable, practical example

```
print('--------------')
print('|            |')
print('|            |')
print('|            |')
print('|            |')
print('--------------')
```

these statements are used to draw a square.

Instead of using these 7 lines again and again to draw a square, lets create a function 'sq'. The function will give same square upon calling.

37 – Even odd function

Let's create some simple functions. write a function that will take one number as parameter and print whether the number is even or odd.

38 – Find the error

Rewrite following functions error free:

```
hello()
def hello(name='Ali Abbas', age= 40, occupation ='Wakeel'):
    print('Ali is a 40 years old wakeel')
```

```
def code(age=40, name, occupation):
    print(name, 'is a', age, 'years old', occupation)
```

```
def movies(name='Lord of rings', part = 3):
    print('Movie', name, 'has' + part, 'parts')
```

```
def person(address, contact, name, organization, name)
    print('thanks for data')
```

39 – Simple code to function

```
import random
for i in range(4):
    num1 = random.randint(1, 10)
    num2 = random.randint(1, 10)
    answer = ''
    while answer != num1*num2:
        answer = int(input(str(num1) + ' X ' + str(num2) + ' = '))
```

Remember this code? This is solution of assignment number 25. See this code and read instructions of assignment 25. Once you get this code,

Let's make a function of this code.

function name: 'mul'.

Parameter: num

In second line the integer within range() will be replaced by 'num'.

Define and call your function to see the results.

40 – Table multiplication, function

Write a simple function that takes one number as parameter and print its multiples from 1 to 10 in logical way.

For example if your given number is 9, output should looks like:

9 X 1 = 9

9 X 2 = 18

9 X 3 = 27

.

.

.

9 X 10 = 90

Section 6: Everything merged

41 – School menu system

Let's write a menu for school system. Our menu system will have option to:

1 - Admit students in school

2 - Remove students from school

3 - View all students of school

Here are some images of output of program for your guidance.

Admission of student


```
In [28]: schoolmenu()

         ============================Menu============================
         1 - Addmission of student in school
         2 - Struck student from school
         3 - View all students of school
         4 - Quit from menu

         Enter your choice: 1

         -----Add student-----

         Enter student name: Ali

         Student Added!
```

Removal of student

```
==============================Menu==============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 2

-----Remove student-----

Enter name of studentAli

Student removed!
                                        Menu
```

Admission of student

```
==============================Menu==============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 1

-----Add student-----

Enter student name: Ahmad

Student Added!
```

Admission of student

```
===============================Menu===============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 1

-----Add student-----

Enter student name: Ali

Student Added!
```

Admission of student

```
===============================Menu===============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 1

-----Add student-----

Enter student name: Abdullah

Student Added!
```

View students

```
==============================Menu==============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 3

----------Students in school---------
Ahmad
Ali
Abdullah
```

Removal of student who is not present in school

```
==============================Menu==============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 2

-----Remove student-----

Enter name of studentHarman
No student found
                              Menu
```

Exit program

```
==============================Menu==============================
1 - Addmission of student in school
2 - Struck student from school
3 - View all students of school
4 - Quit from menu

Enter your choice: 4
=================Good bye===============
```

Furthermore here are some instructions for this project:

1 - Use list to store data (student names).

2 - For task 1, use list built-in method to add student in list.

3 - For task 2, use conditional statement to check whether the student is in school or not. Remove student if he is in school by using list built-in method 'remove()'.

4 - For task 3 use for loop and iterate over the list to display all students.

5 - For implementation of last point, you need to run program again and again unless user decide to exit. For that enclose whole program in while loop such that it should run again and again. Use conditional statement to check whether user want to exit, then you may use break statement to stop while loop.

For our case, we implemented this in a function but you can make it without function as well.

42 – Hospital management system

Our purpose is to

- Hold data of patients. We will store name and age of each of patient.
- Our hospital can only accommodate 40 patients. So if patients are more than 40, we will indicate that we cannot accommodate this much patients.
- Our hospital do not provide services to old age persons. If there is any patient more than 60 years old, we will not add him in hospital data.

Instructions:

- Take input from user about number of patients, then iterate for loop as many times as there are patients. Each time loop runs, ask for data of patient name and age and store that in a list and then append that list in a major list that will at the end contain data of all the patients.
- Use conditional structure to check if age is greater than 60. If it is greater, we won't admit that patient in hospital. It means we will not store data of that patient.
- Now print how many more patients we can accommodate in our hospital.

In case user input number of patients as more than 40, you will have to indicate that we cannot accommodate more than 40 patients, and program should stop further execution.

Here are some sample executions but obviously your statements can be different from mine.

Case 1: Input data smoothly

```
How many patients in hospital: 5

Enter 1 patient name: Ali
Enter 1 patient age: 43

Enter 2 patient name: Ahmad
Enter 2 patient age: 32

Enter 3 patient name: Mujtaba
Enter 3 patient age: 23

Enter 4 patient name: Alexa
Enter 4 patient age: 11

Enter 5 patient name: Harman
Enter 5 patient age: 20
=====================Thanks for prividing this data================================

Total patients:  5
Occomodation available: 40
We can accomodate 35 patients.
```

Case 2: Input data of more than 40 patients

```
In [8]: a()

        How many patients in hospital: 45
        WARNING: we can only accomodate 40 patients

In [ ]:
```

Case 3: Input data of patients of age more than 60 years

```
How many patients in hospital: 3

Enter 1 patient name: Ali
Enter 1 patient age: 59

Enter 2 patient name: ahmad
Enter 2 patient age: 71

Patients above 60 aren't provided service in our hospital



Enter 3 patient name: Huzaifa
Enter 3 patient age: 65

Patients above 60 aren't provided service in our hospital


====================Thanks for prividing this data============================

Total patients:  1
Occomodation available: 40
We can accomodate 39 patients.
```

Good luck!

---

43 – Employee salaries management system

An organization need you to calculates weekly employee payroll for all employees.

For each employee, ask the user for the employee ID number, the hourly wage rate, and the number of hours worked during a week.

Each employee is paid 1.5 times their regular hourly rate for all hours over 40. A tax amount of 3.625 percent of salary is deducted.

At the end print total salary of all employees and number of employees.

INSTRUCTIONS:

- Create two variables with value 0 to store total salary and total number of employees so that we can print that data at the end

- Initialize a while loop, take ID, hourly_wage, hours_worked as inputs and do all the computations to calculate net salary of that employee. This while loop should stop running when user enter 0 as employee id

- Also write code to add salary of employee and employee number in two variable that we created in the first step.

- Once we are out of while loop, program should indicate total salary of all employee and total number of employee.

44 – Regular budget management system

Let's write something functional for someone's budget plan

Its functionality should be:

- Ask user for his expenses continuously unless user enter 0.
- Print total expenses.
- Print whether the person is in Surplus, Deficit, or Balanced.

Surplus: if monthly income is more than expenses

Deficit: if monthly income is less than expenses

Balanced: if monthly income is equal to expenses

- How will you know the monthly income of person?

Instructions

- Create a function 'budget' with a parameter 'available' such that parameter will indicate the total budget of a month.

- Write your code in that function.

- At the end, it should print: total monthly budget, remaining balance, status of person (surplus, deficit, or balanced)

Sample outputs:

Case 1: When expenses are balanced:

```
In [10]: budget(1000)

         Enter an expense (0 to quit): 900
         Enter an expense (0 to quit): 100
         Enter an expense (0 to quit): 0


         Total expense: 1000
         Remaining amount: 0
         Status: Balanced

In [ ]:
```

Case 2: When monthly income is more than expenses

```
In [12]: budget(1000)

         Enter an expense (0 to quit): 500
         Enter another expense (0 to quit): 100
         Enter another expense (0 to quit): 50
         Enter another expense (0 to quit): 0


         Total expense: 650
         Remaining amount: 350
         Status: Surplus

In [ ]:
```

Case 3: When monthly income is less than expenses

```
In [13]: budget(1000)

         Enter an expense (0 to quit): 500
         Enter another expense (0 to quit): 200
         Enter another expense (0 to quit): 700
         Enter another expense (0 to quit): 0


         Total expense: 1400
         Remaining amount: -400
         Status: Deficit

In [ ]:
```

45 – Restaurant menu system

Creating a menu functional menu for restaurant will be a good practice for your logics now.

This exercise will only include function and conditional statement.

Here is some part of code:

```
def res_menu():
    print('===============MENU===============')
```

```python
print('1 - Burger beef - 600')
print('2 - Tortilla Wraps - 800')
print('3 - Chai Paratha - 200')
print('4 - Chai - 150')
print('5 - Shake (Mango, banana, date, chiku) - 600')

menu = [600, 800, 200, 150, 600]
choice = int(input('\n\nEnter your option: '))
```

You will continue this menu to meet these requirements:

- In the last line of upper code, user will enter his choice about what he want to order from the menu, extract prize of that item from the list 'menu' using list indexing.

- Ask user for quantity of item he want to order and then print out total bill

Once done, run your code and analyze the outputs.

Sample run:

```
================MENU================
1 - Burger beef - 600
2 - Tortilla Wraps - 800
3 - Chai Paratha - 200
4 - Chai - 150
5 - Shake (Mango, banana, date, chiku) - 600


Enter your option: 4
Enter quantity: 3

------ Your Total: 450 -------
```

`]:`

After you are done with this, follow these this:

As you see in the menu's 5th option, there are shake of different flavors. Let's refine our code such that if user want to order shake, we will ask him to enter the flavor as well.

This will not change the bill for user but this will make our menu more logical and detailed. You can use conditional statements for this implementation.

Sample run:

```
=================MENU================
1 - Burger beef - 600
2 - Tortilla Wraps - 800
3 - Chai Paratha - 200
4 - Chai - 150
5 - Shake (Mango, banana, date, chiku) - 600


Enter your option: 5


1 - Mango
2 - Banana
3 - Date
4 - chiku
5 - Icecream (chocolate)

Enter you choice of flavor of shake: 3
Enter quantity: 2

------ Your Total: 1200 -------
```

46 – Advance restaurant menu system

Let us refine the code of previous assignment (45)

Previously we were taking only one item in order, but customer may want to order more than one item. For example he may want to order shakes as well chai paratha and maybe more.

Do something such that the program will take order from user again and again until user press zero to stop the order. Then represent total bill.

47 – Food cost management

Write a function that asks a user the costs of breakfast, lunch, and supper for one day. Print total cost after these inputs.

After entering data for each day asks the user whether they want to enter data for another day.

At the end:

Print cost of all the breakfast, all the lunch, and all the supper separately. Also print Total cost of all meals. For this purpose you may need to create few variables in the beginning of function to hold data of all meal costs.

Your statements can be different, but here is a sample output:

```
Enter the cost of breakfast for the day: 43
Enter the cost of lunch for the day: 56
Enter the cost of supper for the day: 43
Your total for the day was 142.0

Do you want to enter data for another day? (yes/no): yes

Enter the cost of breakfast for the day: 34
Enter the cost of lunch for the day: 67
Enter the cost of supper for the day: 43
Your total for the day was 144.0

Do you want to enter data for another day? (yes/no): no
==========================

Breakfast: 77.0
Lunch: 123.0
Suppers: 86.0

All meal cost: 286.0
```

48 – Simple visualizations

Let's plot some graphs. But we didn't covered graphs yet... No worries we will follow something that we already know.

Our task is to

visualize sales in a market through '*'

How

- Let suppose someone own 5 shops in a market. Write a program such that user will input sales of those 5 shops of 1 day. You can do this manually, and you can also use for loops. Store all sales in some variable.

- If sale of a shop is 2500, there will be 25 '*' in the graph. If there will be sale of 800, there will be 8 '*' in the graph.

This is how output should look like

Enter today's sales for store 1: 1000 [Enter]

Enter today's sales for store 2: 1200 [Enter]

Enter today's sales for store 3: 1800 [Enter]

Enter today's sales for store 4: 800 [Enter]

Enter today's sales for store 5: 1900 [Enter]

SALES BAR CHART

Store 1: **********

Store 2: ************

Store 3: **********************

Store 4: ************

Store 5: **********************

- If there are 400 worth sales, you can calculate number of '*' by dividing it by 100: 400 / 100 = 4

- Then you will be able print 4 '*' (for this case) by using: '*' * 4

- Store data in proper variable.

- You will be able to generate these results without using loop structures and that is fine as well.

---

49 – Student marks

In this assignment, we will write a function that will take a list of marks of students in accordance with their roll numbers.

[23, 43] this will indicate that roll number 1 marks are 23 and roll number 2 marks are 43

Marks are out of 50. You will have to make bar charts for each student in the list.

- First print all marks of students with roll number using a for loop.
- Print bar chart of student marks.

Sample outputs:

```
In [14]: st([19, 15, 34])

         Roll number 1 marks: 19
         Roll number 2 marks: 15
         Roll number 3 marks: 34


         ===================BAR CHART===================

         Roll number 1 marks: *******************
         Roll number 2 marks: ***************
         Roll number 3 marks: **********************************
```

Another

```
In [16]: st([46, 25, 37, 49, 12, 31, 23])

Roll number 1 marks: 46
Roll number 2 marks: 25
Roll number 3 marks: 37
Roll number 4 marks: 49
Roll number 5 marks: 12
Roll number 6 marks: 31
Roll number 7 marks: 23


===================BAR CHART=====================

Roll number 1 marks:  **********************************************
Roll number 2 marks:  *************************
Roll number 3 marks:  *************************************
Roll number 4 marks:  *************************************************
Roll number 5 marks:  ************
Roll number 6 marks:  *******************************
Roll number 7 marks:  *********************
```

50 – Rock, paper scissor

Write a program that let the user play the **game of Rock, Paper, Scissors** against the computer. The program should work as follows.

1. When the program begins, create a list ['rock', 'paper', 'scissor'] and let the computer select one of these randomly.
2. The user enters his or her choice of "rock", "paper", or "scissors". (You can use a menu if you prefer.)
3. The computer's choice is displayed.
4. A winner is selected according to the following rules:

- If one player chooses rock and the other player chooses scissors, then rock wins.
- If one player chooses scissors and the other player chooses paper, then scissors wins.
- If one player chooses paper and the other player chooses rock, then paper wins.
- If both players make the same choice, it is a tie

Use as many conditional statements as you want.

Do all the implementations in function so that we can play game again and again.

Congratulations