MITS5004 IT Security Assignment Help

Table of Contents

Fundamentals of Programming

Introduction

I am working as a junior software developer in a financial organization. The organization has been tasked with developing employee income based on the number of hours worked. This report will provide a comprehensive analysis of the fundamentals of programming and its associated concepts. In the process of developing the employee wage calculation application, I am tasked with using the Python 3 application. The program should involve the calculation of income tax, superannuation, deductions, and other additional elements related to wages. This report will provide all information regarding the development process, pseudocode, and flowchart involved in the development of functional software applications for this financial organization. This provides complete information on high-level programming models and support in implementing fields of programming concepts as required.

Pseudocode and Flow Chart

Pseudocode

The pseudocode is created for the entire income calculation program.

```
➢ Step 1: Get User Input of Employee Name
➢ Step 2: Get User Input of HoursWorked and HourlyRate
➢ Step 3: Check whether the input valid or not
    o If input is not valid prompt to get user input
    o Else go to Step 4
➢ Step 4: Calculate Income
        ▪ $Income = HoursWorked * HourlyRate$
➢ Step 5: Calculate IncomeTax Deduction with 20% Tax rate
        ▪ $IncomeTax = income * taxrate$
➢ Step 6: Calculate Superannuation Deduction with 10% Superannuation rate
        ▪ $Superannuation = income * Superannuation rate$
➢ Step 7: Print the Result
```

The pseudocode provides all the key steps involved in the program from the input collection, calculation of income and performing taxation and etc. It also provides information that the output displays to the user. The user can able to adjust the program according to their needs.

Flow Chart

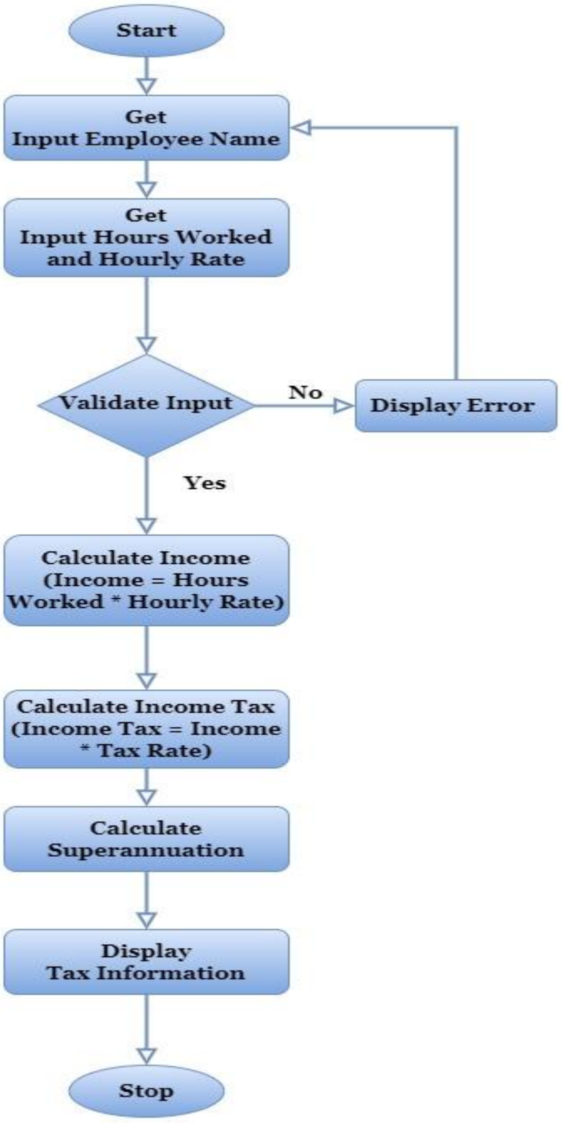


Figure 1 Flowchart for program

The above given flowchart uses standardized flow in dealing with the given case information. From the algorithm developed, all the key steps and their interaction on program is provided. The users or developers can able to spell out the logic behind program and to automate their entire process (Joshuva, 2022). This shows overall relationship between input, process, and output of the program. This will helpful for all the stakeholder to share ideas and communicate with them. The employee wages calculating program focuses on addressing all the inputs, process and outputs. In overall the visual representation of the program function is clearly depicted in the flowchart.

Program Implementation

Python Code

taxcalculation.py

```
def calculateIncomeTax(Income):

    #income tax deduction Calculation based on the given income

    tax_rate = 20/100   #20% is taxed

    return tax_rate * Income



def calculateSuperannuation(Income):

    #superannuation deduction Calculation based on the given income

        super_rate = 10/100 # Superannuation rate is 10%

    return super_rate * Income

def printTaxInfo(EmployeeName, Income, IncomeTax, Superannuation):

    #Display Result

    print(" \n--------------------------------------------------------------------------------")

    print("| Employee Name | Income     | Income Tax Deduction | Superannuation Deduction |")

    print("|---------------|------------|----------------------|-------------------------|")

    print(f"|           {EmployeeName:<14}|           ${Income:<10.2f}|           ${IncomeTax:<20.2f}|
${Superannuation:<24.2f}|")
```

```
    print(" ------------------------------------------------------------------------------")
def main():


    while True:


        try:
            #Get User input of employee name
            EmployeeName = input("\nEnter Employee Name: ")


            #Get User input of HoursWorked
            HoursWorked = float(input("\nEnter Hours Worked: "))


            #Get User input of HourlyRate
            HourlyRate = float(input("\nEnter Hourly Rate: "))


            if HoursWorked < 0 or HourlyRate < 0:
                raise ValueError("Hours worked and hourly rate must be Positive number")


        except ValueError:
            #print error message
            print("\nInvalid input. Please enter Valid Input")
            continue
        else:
            break
       #calculate the income
    Income = HoursWorked * HourlyRate
        #calculate the IncomeTax
    IncomeTax = calculateIncomeTax(Income)
        #calculate the Superannuation
    Superannuation = calculateSuperannuation(Income)
    #Print Employee Tax Information
    printTaxInfo(EmployeeName, Income, IncomeTax, Superannuation)
if __name__ == "__main__":
    main()
```

Thus the taxcalculation.py is developed. The program is developed by applying all the concepts of python and logically used all indentation. The calculate income tax and calculate superannuation function are used for calculating the deduction based on the user income. This will helpful in maintaining the taxation rate for each employee (Joshuva, 2022). All the taxation function was focusing on employee tax information gathering overall information.

In the main function of the program, the prompt will collect input from the user and to perform calculation. There were several loops are provided within the function and to handle it effectively with input validation. The wage calculation program ensures number of hour workers and to manage their negative model. The program will perform overall calculation of income, income tax and superannuation (Gandhi, 2021). Then it focuses on passing information and to display the results effectively. The loop can able to handle if the user input us negative and arise errors. In overall the program have well-structured with input validation, functional modularity and formatted output.

Input Validation

In the developed program, all the inputs are validated using specific function. The while, try, input method, float of input method, value error, continue and else block method are utilized.

- The while is used for creating infinite loop for providing prompt and perform validation.
- The try block is used to execute the code effectively.
- In this input function, users can able to provide their personal information through prompt.
- The float used on the conversion of user input will able to manage the exception raises from user side.
- The exceptional value error block is implemented to manage the catches and prints error message.
- Continue is used for starting the loop and perform next iterations.
- Else is used for blocking the complete one without raising the exception.

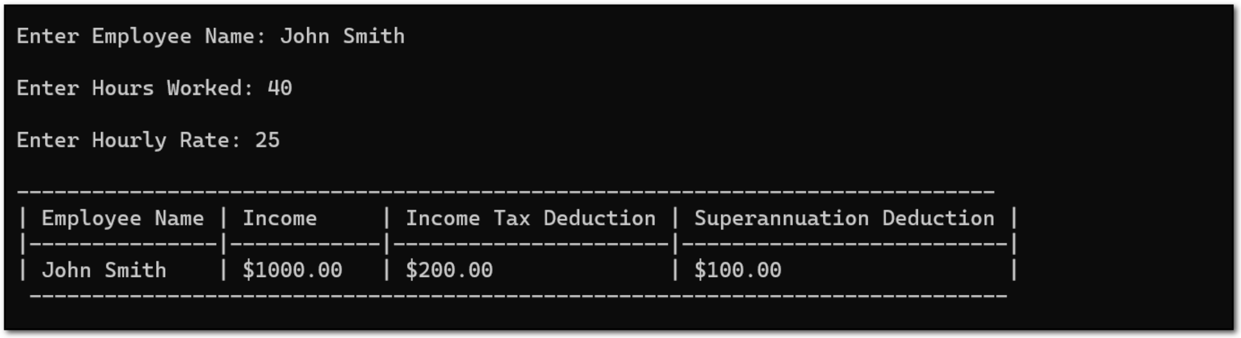
Coding Standards

In the python program, we have used the functional mourality principle. There are totally 4 methods such as calculateIncomeTax, calculateSuperannuation, printTaxInfo, and main are used for promoting modularity and readability of texts. All the variables are clearly labelled that shows that the program is well created. The variable names were utilized such as EmployeeName, HoursWorked, HourlyRate, Income, IncomeTax, and Superannuation.

The program provides detailed outcome using formatted strings. The presentation f output is managed to show the wage standards in table format. The main function is established with conditional statement for execution of modular code.

Testing

Test case 1



```
Enter Employee Name: John Smith

Enter Hours Worked: 40

Enter Hourly Rate: 25

-------------------------------------------------------------------------------
| Employee Name | Income     | Income Tax Deduction | Superannuation Deduction |
|---------------|------------|----------------------|--------------------------|
| John Smith    | $1000.00   | $200.00              | $100.00                  |
 -------------------------------------------------------------------------------
```

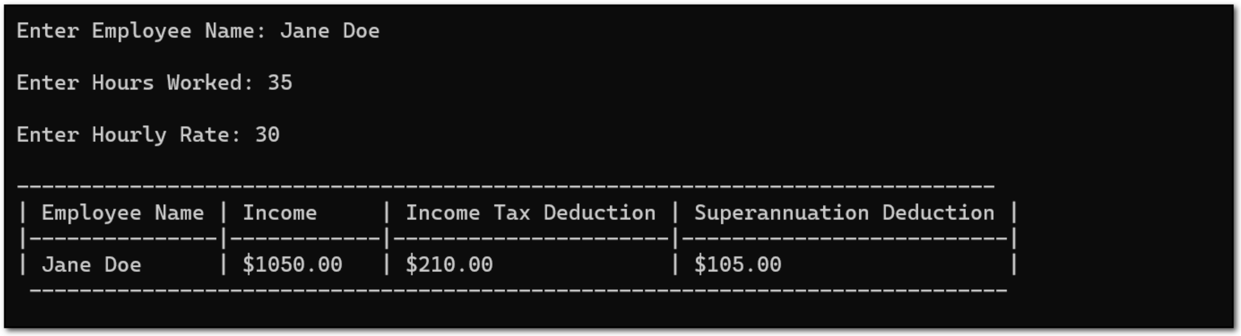
Input:

I have provided input for employee name, hours woeers and hourly rate of that employee as mentioned in the screen.

Expected Output:

It is expected to provide the income, income tax deduction, and superannuation deduction in a formatted table.

Test case 2



```
Enter Employee Name: Jane Doe

Enter Hours Worked: 35

Enter Hourly Rate: 30

-------------------------------------------------------------------------------
| Employee Name | Income     | Income Tax Deduction | Superannuation Deduction |
|---------------|------------|----------------------|--------------------------|
| Jane Doe      | $1050.00   | $210.00              | $105.00                  |
 -------------------------------------------------------------------------------
```

Similarly, the second test case is also performed. Similarly, the information have given and it seems that the program calculated accurately.

Expected output

The program should display valid output and passed the test case.

References

Gandhi, P. (2021, September 30). *Python programming concepts that made my code efficient!* Medium. https://towardsdatascience.com/python-programming-concepts-that-made-my-code-efficient-68f92f8a39d0

Joshuva. (2022). *Object-Oriented Programming (OOP) in Python 3*. Learn Data Science and AI Online | DataCamp. https://www.datacamp.com/blog/how-to-learn-python-expert-guide