Module Title

Fundamentals of Data Science

Assessment Weightage & Type

Weekly Assignment 1 and 2 - Coursework & Regular

Year

2025

Student Name: NIRVIK K.C.

UWE ID: 25024649

Assignment Due Date: July 7, 2025

Assignment Submission Date: July 7, 2025

Bi-weekly assignment

Module Details

Module Code	**UFCFK1-15-0**
Module Title	**Fundamentals of Data Science**
Module Tutors	**Saurav Gautam**
Year	**2024-2025**
Component/Element Number	**PSA/Bi-weekly assignment/Regular**
Weighting	**10%**

Dates

Submission Date	**07-July-2025**
Submission Place	**Backboard**
Submission Time	**23:59**
Submission Notes	**Submit Gitlab URL**

Assignment 1

This assignment consists of the programming questions related to the topics of week 1 and week 2. The main topics of questions are: Python Basics, Operators, and Conditional Statements.

All the students are required to follow the format of the program as specified in the guideline below.

1. All the programs should have initial **doc string** comment (‘’’ description of program‘’’) mentioning what your program will do.
2. Try to maintain single/multi-line comments in the place where needed to make the program understandable.
3. Maintain proper indention and newline spaces to increase the readability of the program.
4. The deliverable are 2 type of files (a single word file and multiple python program files):

a) Separate python program files with **.py** extension (e.g. program_name.py). Provide a relevant name to your program file on the basis of functionality of the program.
b) A word file describing the working of all the programs according to their number. The details required in this is the description of program, screenshot of the testing (input given and output obtained in the execution environment such as IDLE or Command prompt or terminal whichever you prefer.). It is preferred that you work with multiple inputs and outputs.

Questions

1. Create a diagram to show how python works. The diagram should show the components such as input, processing and output.

Answer :

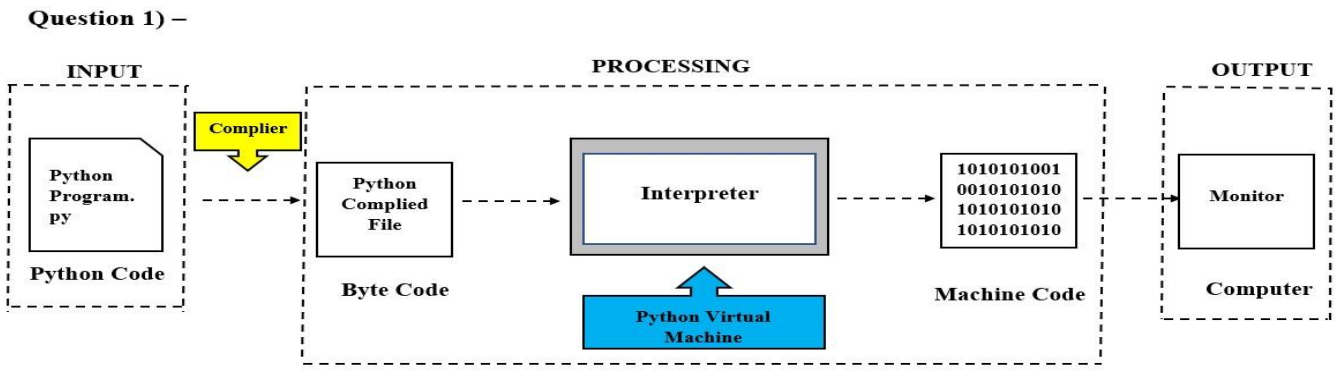


Figure : The diagram that show the working of python with components such as input, processing and output.

2. Write a program to input 2 numbers from the users and display the output of below mentioned operations in a proper format.

I. Addition
II. Subtraction
III. Multiplication
IV. Division
V. Modulo division
VI. Floor division

I. Addition

Answer :

The given python code below asks the user for two numbers and then performs addition between the two numbers to display the result.

Following code for input :

Addition of two numbers as input from the user.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II...

File Edit Format Run Options Window Help

```
'''
Program takes two numbers as input from the user and displays the result of their addition.
'''
# Function to perform addition
def add_numbers(a, b):
    return a + b

# Input from user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

# Perform addition and store result
result = add_numbers(num1, num2)

# Display output
print(f"\nThe result of adding {num1} and {num2} is: {result}")
```

Output obtained in execution :

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bi
t (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester
II/FODS - Weekly Task/Weekly Assigment - 1 and 2/Python files/Addition of two
numbers as input from the user.py
Enter the first number: 5
Enter the second number: 10

The result of adding 5.0 and 10.0 is: 15.0
>>>
```

Python Program File: Saved the above code in a file named “Addition of two numbers as input from the users.py.”

Explanation of code :

User input () :

The program prompts the user to input two numbers using the input() function. The function reads input as a string and changes it to a floating point.

Variable Definition :

We have two variables stored as ‘num1’ and ‘num2’ in the program. The add_numbers(a, b) function takes two parameters and returns their sum.

Addition Calculation :

Finally, the program calculates the addition between the two numbers and stores it in a variable called result.

Output of the program :

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to gather user input and perform addition in python. First, we must set up the variables, utilize the addition operator (+), and display the results with the print() function.

II. Subtraction

Answer:

The given python code below asks the user for two numbers and then performs difference between the two numbers to display the result.

Following code for input :

Subtraction of two numbers as input from the user.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/F...

File Edit Format Run Options Window Help

```
'''
Program takes two numbers as input from the user and displays the result of their subtraction.
'''

# Function to perform subtraction
def subtract_numbers(a, b):
    return a - b

# Input from user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

# Perform subtraction between two numbers
result = subtract_numbers(num1, num2)

# Display result
print(f"\nThe result of subtracting {num2} from {num1} is: {result}")
```

Output obtained in execution :

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bi
t (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester
II/FODS - Weekly Task/Weekly Assigment - 1 and 2/Python files/Subtraction of
two numbers as input from the user.py
Enter the first number: 10
Enter the second number: 15

The result of subtracting 15.0 from 10.0 is: -5.0
>>>
```

Python Program File: Saved the above code in a file named “Subtraction of two numbers as input from the users.py.”

Explanation of code :

User input ():

The program prompts the user to input two numbers using the input() function. The function reads input as a string and changes it to a floating point.

Variable Definition:

We have two variables stored as ‘num1’ and ‘num2’ in the program. The subtract_numbers(a, b) function takes two parameters and returns their difference.

Subtraction Calculation:

Finally, the program calculates the subtraction between the two numbers and stores it in a variable called result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

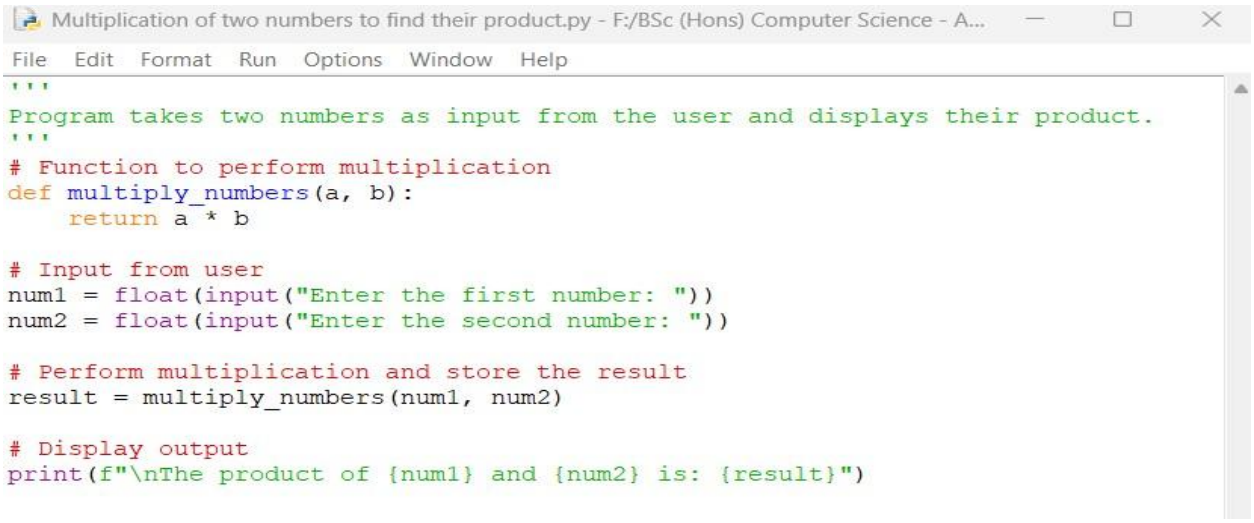
The given program is an example of how to gather user input and perform subtraction in python. First, we must set up the variables, utilize the subtraction operator (-), and display the results with the print() function.

III. Multiplication

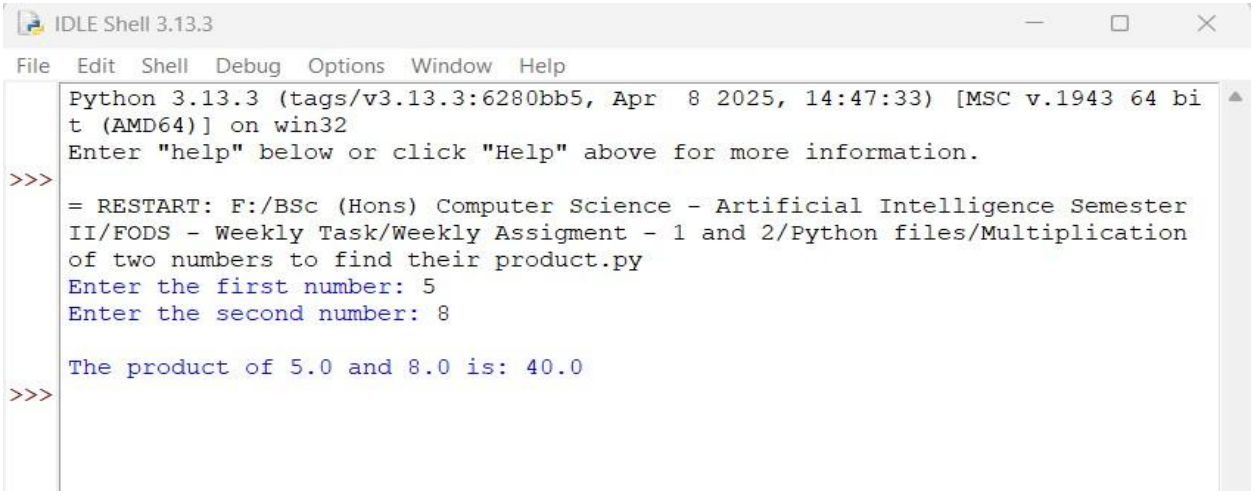
Answer:

The given python code below asks the user for two numbers and then performs multiplication between the two numbers to display the result.

Following code for input :



Output obtained in execution :



Python Program File: Saved the above code in a file named “Multiplication of two numbers to find their product.py.”

Explanation of code :

User input ():

The program prompts the user to input two numbers using the input() function. The function reads input as a string and changes it to a floating point.

Variable Definition:

We have two variables stored as ‘num1’ and ‘num2’ in the program. The subtract_numbers(a, b) function takes two parameters and returns their product.

Multiplication Calculation:

Finally, the program calculates the product between the two numbers and stores it in a variable called result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

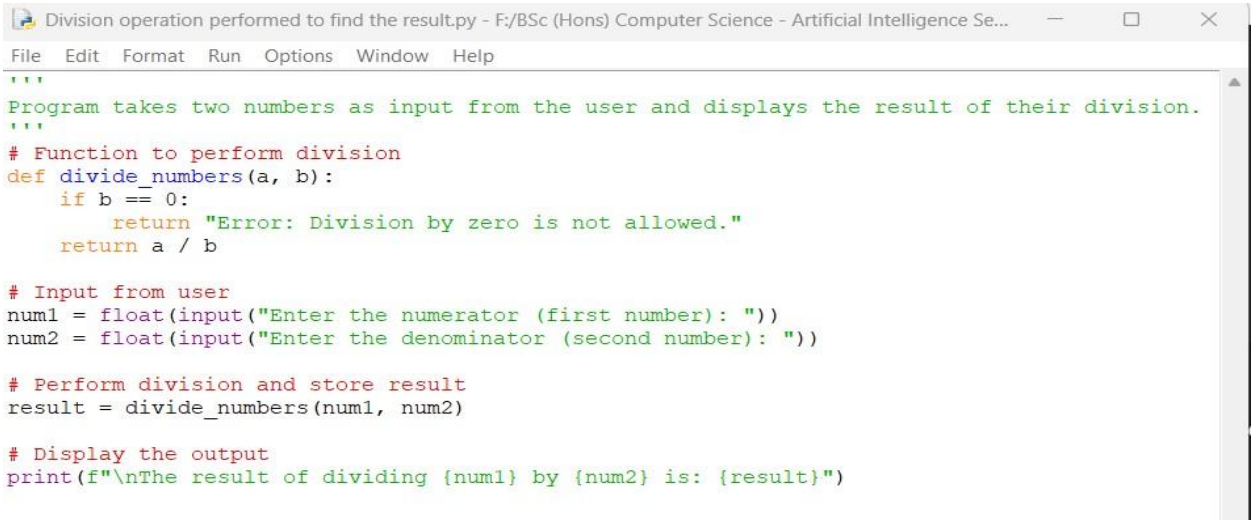
The given program is an example of how to gather user input and perform multiplication in python. First, we must set up the variables, utilize the multiplication operator (*), and display the results with the print() function.

IV. Division

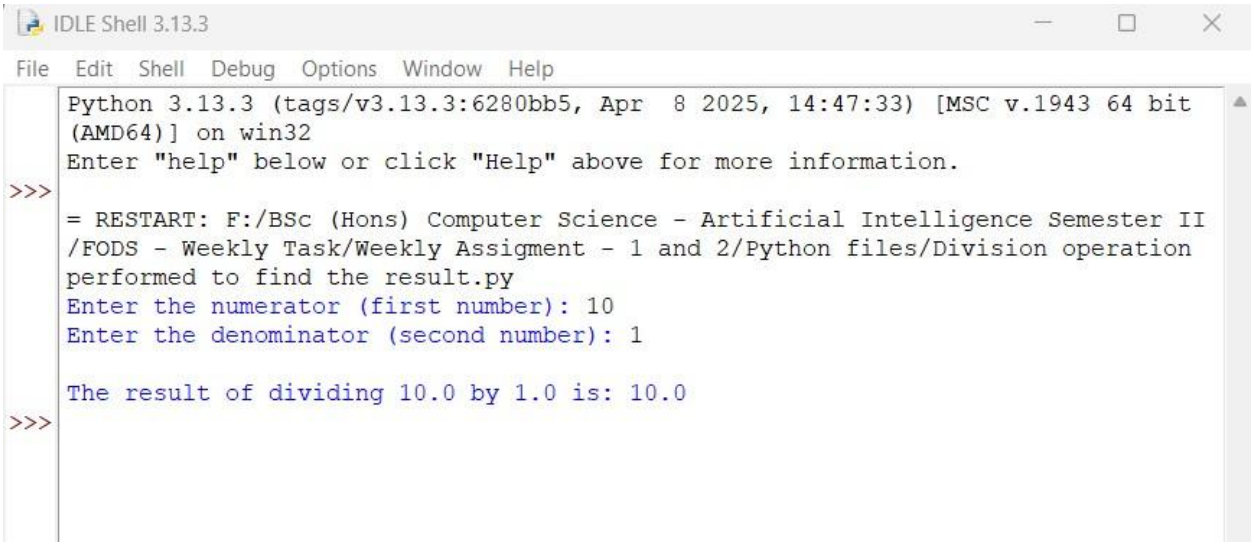
Answer:

The given python code below asks the user for two numbers and then performs division between the two numbers to display the result.

Following code for input:



Output obtained in execution:



Python Program File: Saved the above code in a file named “Division operation performed to find the result.py.”

Explanation of code:

User input ():

The program prompts the user to input two numbers using the input() function. The function reads input as a string and changes it to a floating point.

Variable Definition:

We have two variables stored as ‘num1’ and ‘num2’ in the program. The subtract_numbers(a, b) function takes two parameters and returns their quotient.

Multiplication Calculation:

Finally, the program calculates the division between the two numbers and stores it in a variable called result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to gather user input and perform division in python. First, we must set up the variables, utilize the division operator (/), and display the results with the print() function.

VI. Floor Division

Answer:

The given python code below asks the user for two numbers and then performs floor division between the two numbers. The largest integer that is less or equal to the result of the division is displayed as the result.

Following code for input:

Floor division perfomed to find the result.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS - Weekl...

File Edit Format Run Options Window Help

```
'''
Program takes two numbers as input from the user and displays the result of their floor division.
'''
# Function to perform floor division
def floor_divide_numbers(a, b):
    if b == 0:
        return "Error: Division by zero is not allowed."
    return a // b

# Input from user
num1 = float(input("Enter the numerator (first number): "))
num2 = float(input("Enter the denominator (second number): "))

# Perform floor division and store result
result = floor_divide_numbers(num1, num2)

# Display output
print(f"\nThe result of floor dividing {num1} by {num2} is: {result}")
```

Output obtained in execution:

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bi
t (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester
II/FODS - Weekly Task/Weekly Assigment - 1 and 2/Python files/Floor division
perfomed to find the result.py
Enter the numerator (first number): 10
Enter the denominator (second number): 3

The result of floor dividing 10.0 by 3.0 is: 3.0
>>>
```

Python Program File: Saved the above code in a file named “Floor division performed to find the result.py.”

Explanation of code:

User input ():

The program prompts the user to input two numbers using the input() function. The function reads input as a string and changes it to a floating point.

Variable Definition:

We have two variables stored as ‘num1’ and ‘num2’ in the program. The subtract_numbers(a, b) function takes two parameters and returns their quotient.

Multiplication Calculation:

Finally, the program calculates the floor division between the two numbers and stores it in a variable called result. Like in the example given above, 10 is divided by 3 which gives 3.333. The floor division of 10 by 3 is 3.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

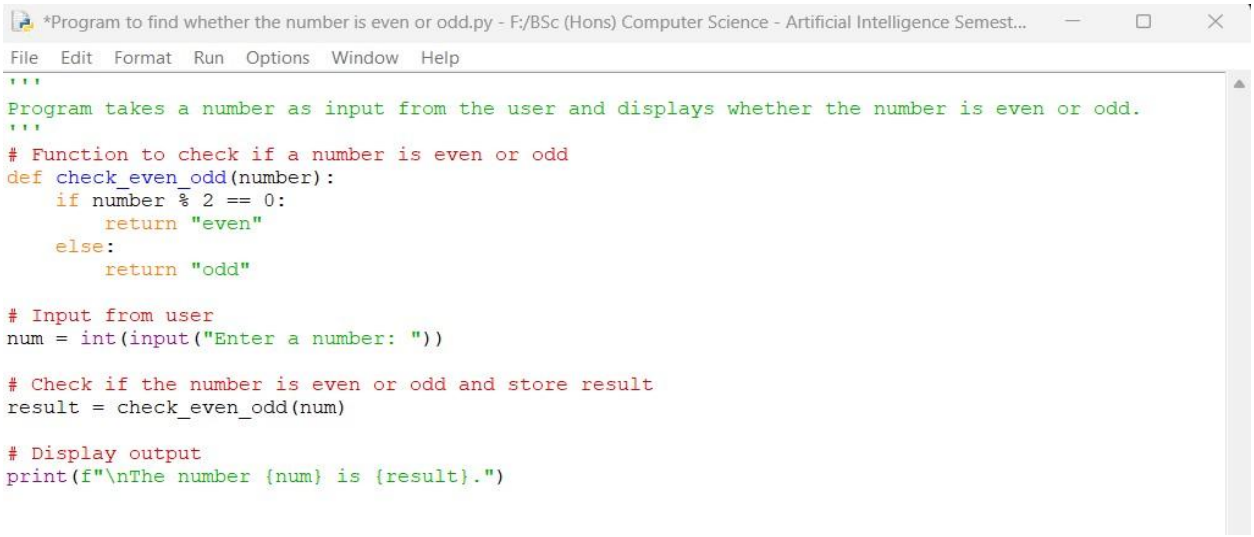
The given program is an example of how to gather user input and perform floor division in python. First, we must set up the variables, utilize the floor division operator (//), and display the results with the print() function.

3. Write a program to take a number input from the user and display whether the number is even or odd.

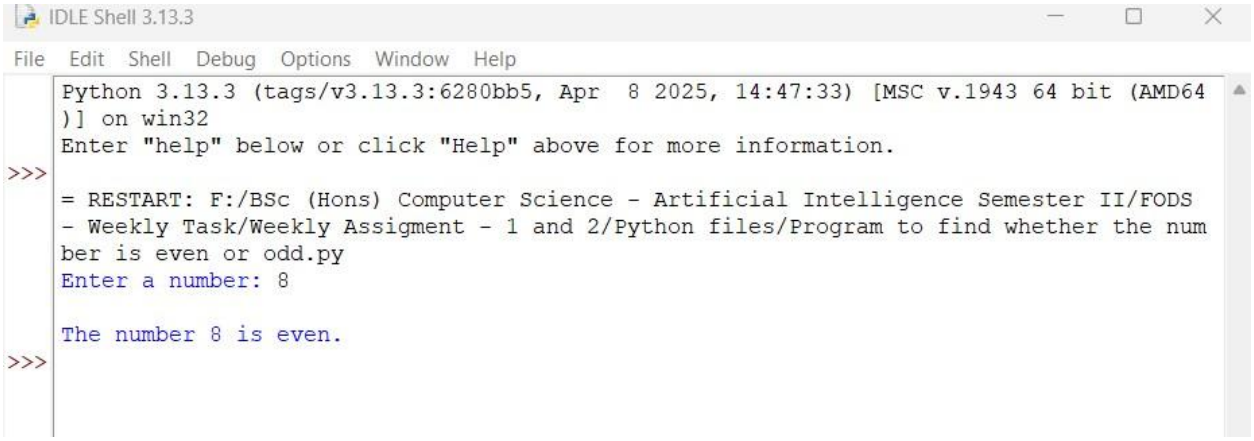
Answer:

The given python code below determines whether the given number by the user is even or odd.

Following code for input:



Output obtained in execution:



Python Program File: Saved the above code in a file named “Program to find whether the number is even or odd.py.”

Explanation of code:

User input ():

The program prompts the user to enter a number using int() function. The input is then converted into an integer using int() which helps to perform arithmetic operations.

Variable Definition:

The ‘num’ variable stores the user provided number. The ‘result’ variable stores the output of even or odd which is checked by ‘check_even_odd’ function.

Even or odd function:

The function takes one parameter ‘number’ and checks if the number is divisible by 2 or not by using modulus operator (%). If number % 2 == 0 then it returns “even” otherwise “odd”.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to determine whether the given number is even or odd in python. First, we must set up the variables, create a function to check even or odd, using of modulus operator (%), and display the results with the print() function.

4. Write a program to take a number input from the user and display the result of some mathematical calculations as mentioned below.

I. Square of the number

Answer:

The given python code below takes a number as input from the user and display the square of that particular number.

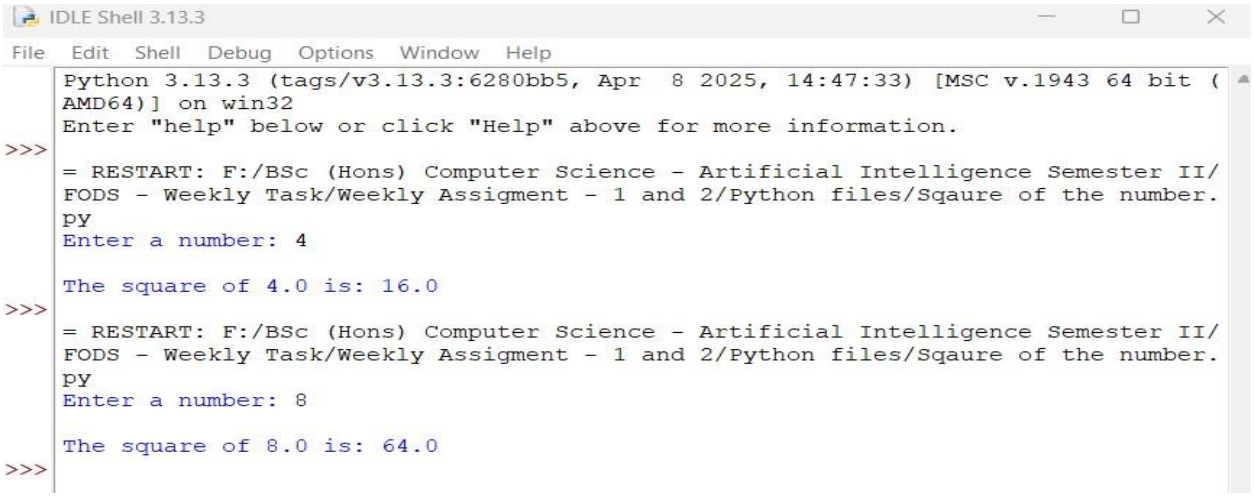
Following code for input:

*Sqaure of the number.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semeste...

File Edit Format Run Options Window Help

```
'''
Program takes a number as input from the user and displays the square of that nu
'''
# Function to calculate the square of a number
def calculate_square(number):
    return number ** 2

# Input from user
num = float(input("Enter a number: "))

# Calculate the square and store result
result = calculate_square(num)

# Display output of the program
print(f"\nThe square of {num} is: {result}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named “Square of the number.py.”

Explanation of code:

User input ():

The program prompts the user to input a number. It is then converted to a floating-point using float(), which helps to handle both integer and decimal inputs.

Variable Definition:

The ‘num’ variable stores the number provided by the user. The ‘result’ variable stores the output of the square calculation which is performed by the ‘calculate_sqaure’ function.

Square Calculation Function:

The ‘calculate_sqaure’ function takes one parameter ‘number’ as input and then calculates the square of that number using exponentiation operator ‘**’. This raises the number to the power of 2.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to find the square the given number to the power of 2 in python. First, we must set up the variables, create a function to perform square calculation using of modulus operator (**) for calculation, and display the results with the print() function.

II. Square root of the number

Answer:

The given python code below takes a input from the user and displays the square of that number.

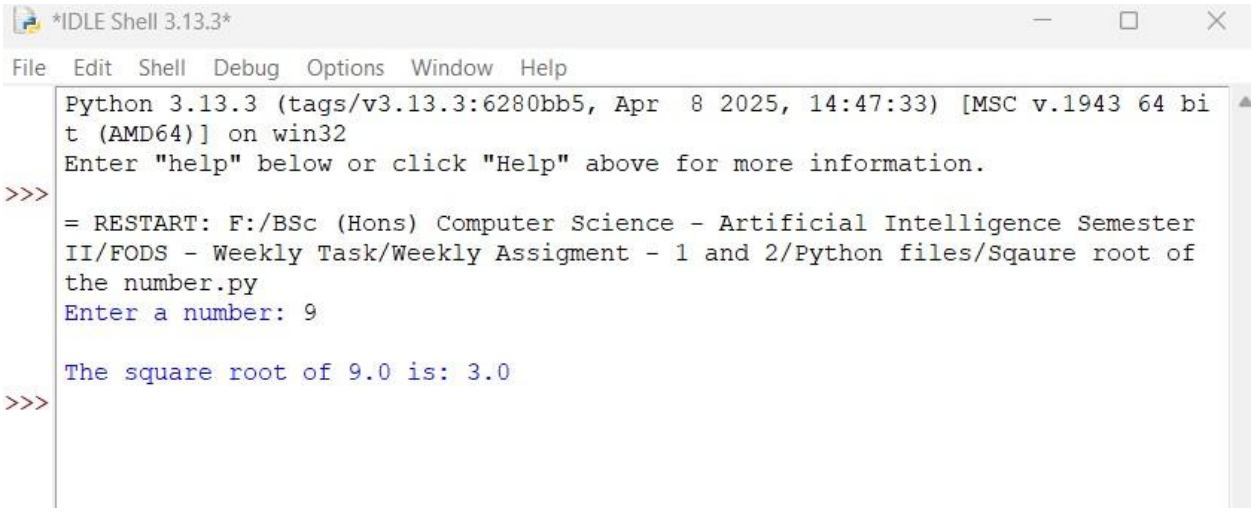
Following code for input:

*Sqaure root of the number.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS -...

File Edit Format Run Options Window Help

```
'''
Program takes a number as input from the user and displays the square root of that number.
'''
import math  # Importing the built-in math module to use the sqrt function

# Function to calculate the square root of a number

def calculate_square_root(number):
    if number < 0:
        return "Error: Cannot calculate the square root of a negative number."
    return math.sqrt(number)

# Input from user
num = float(input("Enter a number: "))

# Calculate the square root and store result
result = calculate_square_root(num)

# Display output of the program
print(f"\nThe square root of {num} is: {result}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named “Square root of the number.py.”

Explanation of code:

User input ():

The program prompts user to enter a number using input() function. The input is read as a string which is converted to a floating-point number using float() function.

Variable Definition:

The ‘num’ is a variable that stores the number provided by the user. The ‘result’ variable stores the output of the square root calculation which is performed by the ‘calculate_sqaure_root’ function.

Square Root Calculation Function:

The function ‘calculate_sqaure_root’ takes one parameter, ‘number’. It checks if the given number is negative If it is, the function returns an error message since the square root of a negative number is not defined in real numbers. If the number is non-negative, it calculates the square root using ‘math.sqrt()’.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

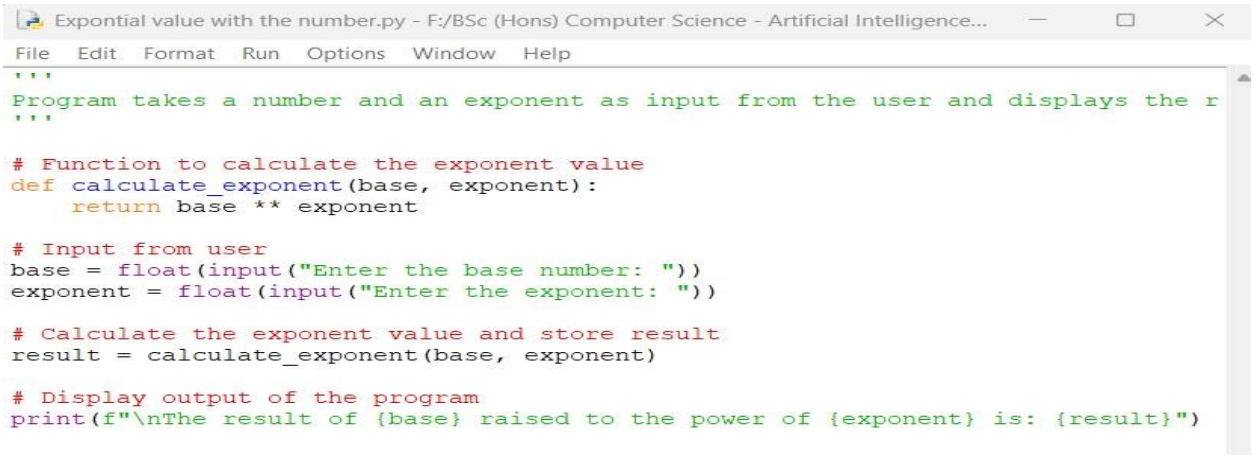
The given program is an example of how to find the square root of the given number in python. First, we must set up the variables, create a function to perform square root calculation, using built-in math module to use sqrt function, error handling if the number is less than 0, and display the results with the print() function.

III. Exponent value with the number

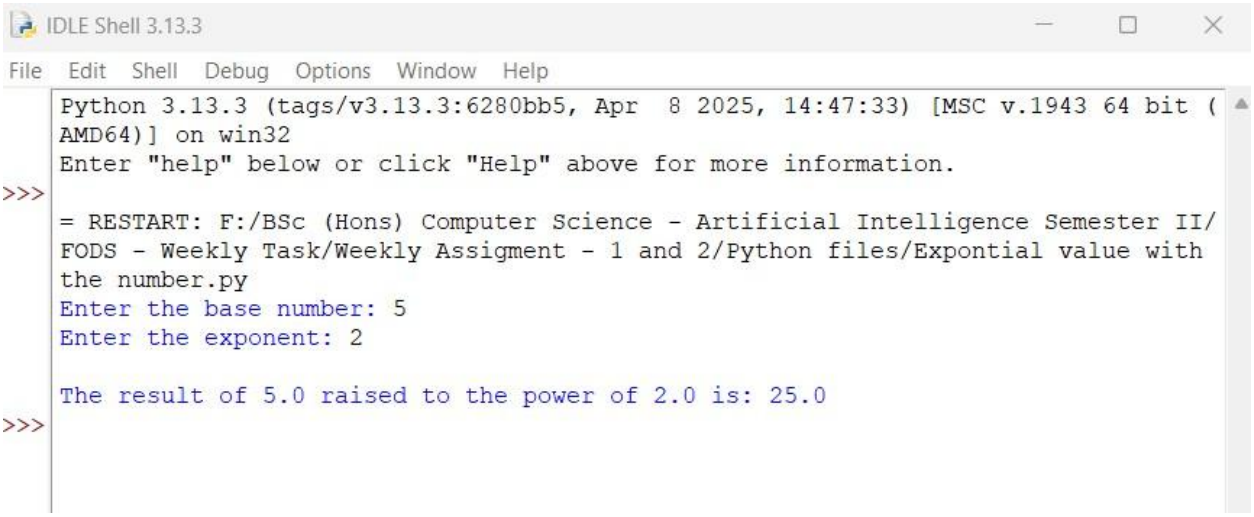
Answer:

The given python code below program takes an input number and an exponent number from the user and displays the result of raising the number to the specified exponent.

Following code for input:



Output obtained in execution:



Python Program File: Saved the above code in a file named “Exponent value with the number.py.”

Explanation of code:

User input ():

The program prompts the user to input base and exponent numbers. Both are converted to floating-point numbers using float().

Variable Definition:

The ‘base’ variable that stores the user-provided base number and ‘exponent’ variable stores the exponent provided by the user. Another variable ‘result’ stores the output of the exponent calculation performed by the ‘calculate_exponent’ function.

Exponent Calculation Function:

The ‘calcualate_exponent’ function takes two parameters: ‘base’ and ‘exponent’. The exponent is calculated using the exponentiation operator ‘**’ which raises the base to the power of exponent.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to find the exponent value when use provided number and exponent value in python. First, we must set up the variables, create a function to perform exponent calculation, using the exponentiation operator (**) for calculation, and display the results with the print() function.

IV. Log Base 10 of the number

Answer:

The given python code below takes a number from the user and displays the result of the logarithm base 10 of that particular number.

Following code for input:

Log base 10 of the number.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS - Weekly Task...

File Edit Format Run Options Window Help

```
'''
Program takes a number as input from the user and displays the logarithm base 10 of that number.
'''

import math  # Importing the built-in math module to use the log10 function

# Function to calculate the logarithm base 10 of a number
def calculate_log_base_10(number):
    if number <= 0:
        return "Error: Logarithm is not defined for non-positive numbers."
    return math.log10(number)

# Input from user
num = float(input("Enter a number: "))

# Calculate the logarithm base 10 and store result
result = calculate_log_base_10(num)

# Display output of the program
print(f"\nThe logarithm base 10 of {num} is: {result}")
```

Output obtained in execution:

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bit (
AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/
FODS - Weekly Task/Weekly Assigment - 1 and 2/Python files/Log base 10 of the nu
mber.py
Enter a number: 10

The logarithm base 10 of 10.0 is: 1.0
>>>
```

Python Program File: Saved the above code in a file named “Log base 10 of the number.py.”

Explanation of code:

User input():

The program prompts the user to enter a number using input() function. The input is read as a string and converted into a floating point using ‘float()’. The program imports built-in math module which provided access to mathematical functions like log().

Variable Definition:

The ‘num’ variable stores the number provided by the user. The ‘result’ variable stores the output of the logarithm calculation performed by the ‘calculation_log_base_10’ function.

Logarithm Calculation Function:

The ‘calculate_log_base_10’ function takes one parameter given by user, ‘number’. It checks the number if the number is less than or equal to zero. If it is, the function returns an error message since the logarithm cannot be defined for non-positive numbers. If the input number is positive, then it calculates the logarithm base 10 using math.log10().

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to find the logarithm value with base 10 using $\log_{10}()$ in python. First, we must set up the variables, import built-in math module to access mathematical create a function to perform logarithm calculation, error handling for non-positive numbers, and display the results with the print() function.

V. Calculate the power 3, 4 and 5 of the number.

Answer :
The given python code below takes a number as input from the user and displays the result of raising that number to the powers of 3, 4, and 5.

Following code for input:

Calculate the power 3, 4, and 5 of the number.py - F:/BSc (Hons) Computer Science - Ar...

File Edit Format Run Options Window Help

```
'''
Program takes a number as input from the user
and displays the results of raising the number to the powers of 3, 4, and 5.
'''
# Function to calculate the powers of a number
def calculate_powers(number):
    power_3 = number ** 3
    power_4 = number ** 4
    power_5 = number ** 5
    return power_3, power_4, power_5

# Input from user
num = float(input("Enter a number: "))

# Calculate the powers and store results
power_3, power_4, power_5 = calculate_powers(num)

# Display output
print(f"\nThe number {num} raised to the power of 3 is: {power_3}")
print(f"The number {num} raised to the power of 4 is: {power_4}")
print(f"The number {num} raised to the power of 5 is: {power_5}")
```

Output obtained in execution:

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS - Weekly Task/Weekly Assigment - 1 and 2/Python files/Calculate the power 3, 4, and 5 of the number.py
Enter a number: 2

The number 2.0 raised to the power of 3 is: 8.0
The number 2.0 raised to the power of 4 is: 16.0
The number 2.0 raised to the power of 5 is: 32.0
>>>
```

Python Program File: Saved the above code in a file named “Calculate the power of 3, 4, and 5 of the number.py.”

Explanation of code:

User input():

The program prompts the user to enter a number using input() function. The input is then converted to a floating-point using float().

Variable Definition:

The ‘num’ variable stores the number provided by the user. Then, the results of power calculation are stored in the variables ‘power_3’, ‘power_4’ and ‘power_5’.

Power Calculation Function:

The ‘calculation powers function takes one parameter from the user, ‘number’. Then, it calculates the powers of the number using the exponentiation operator ‘**’. The variables like ‘power_3’ is calculated as ‘number ** 3’. Similarly, other variables are calculated as well. The function returns the three calculated powers.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to calculate the following power of a value when provided with a number in python. First, we must set up the variables for input number and powers, create a function to perform power calculation, using the exponentiation operator (**) to calculate, and display the results with the print() function.

5. Solve the below mentioned expressions in a python program. Feel free to take input of the required variables to solve the expressions.

I. $a^2 + 2ab + b^2$

Answer:

The given python code below calculates the expression $a^2 + 2^{ab} + b^2$ which is expanded in form as $(a + b)^2$.

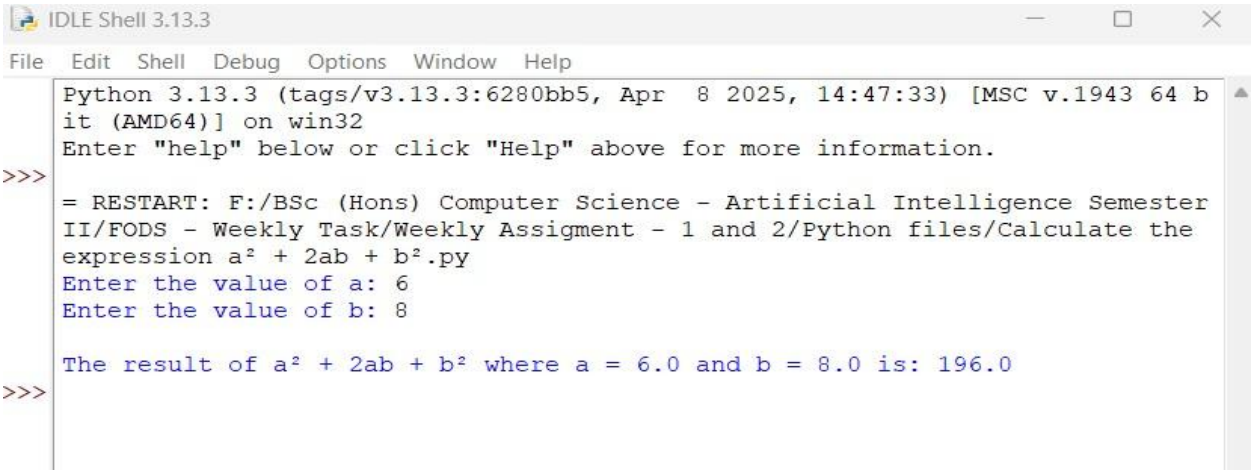
Following code for input:

Calculate the expression a² + 2ab + b².py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/F...

File Edit Format Run Options Window Help

```
'''
Program calculates the expression a² + 2ab + b² by taking inputs for variables a and b.
'''
# Function to calculate the expression
def calculate_expression(a, b):
    return a**2 + 2*a*b + b**2

# Input from user
a = float(input("Enter the value of a: "))
b = float(input("Enter the value of b: "))

# Calculate the expression
result = calculate_expression(a, b)

# Display output
print(f"\nThe result of a² + 2ab + b² where a = {a} and b = {b} is: {result}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named "Calculate the expression $a^2 + 2^{ab} + b^2$.py."

Explanation of code:

User input():

The program prompts the user to enter the values for 'a' and 'b' using the input() function. Both values are converted to floating-point numbers using float().

Variable Definition:

Values 'a' and 'b' store the values provided by the user. The variable 'result' stores the calculated value of the expression.

Expression Calculation:

The expression $'a^2 + 2^{ab} + b^{2'}$ is calculated using process like 'a**2' for a^2, '$2 * a * b$' for 2ab, and 'b**2' for b^2. The function 'calculate_expression()' performs this calculation and returns the result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to calculate the following expressions value when two values are entered as input in python. First, we must set up the variables for input number, create a function to perform expression calculation, using the exponentiation operator (**) to calculate the expressions, and display the results with the print() function.

II. $a^5 + 2abc + b^3 + c^4$

Answer:

The given python code below calculates the expression $a^5 + 2abc + b^3 + c^4$ by taking inputs for the variables a, b, and c.

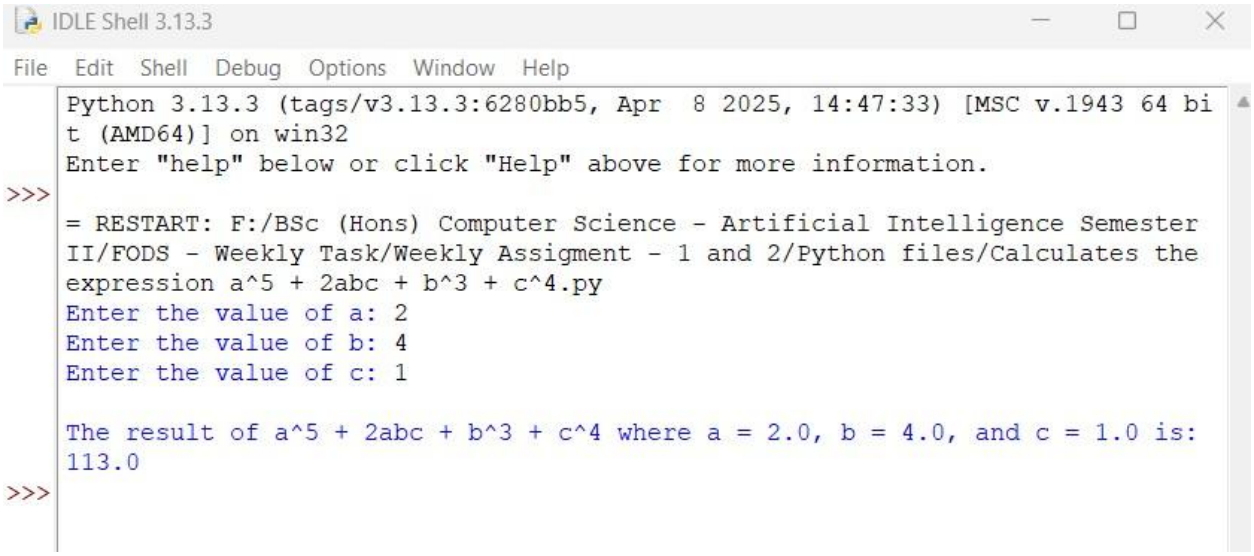
Following code for input:

Calculates the expression a^5 + 2abc + b^3 + c^4.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS -...

File Edit Format Run Options Window Help

```
'''
Program calculates the expression a^5 + 2abc + b^3 + c^4 by taking inputs for variables a, b, and c.
'''
# Function to calculate the expression
def calculate_expression(a, b, c):
    return a**5 + 2*a*b*c + b**3 + c**4

# Input from user
a = float(input("Enter the value of a: "))
b = float(input("Enter the value of b: "))
c = float(input("Enter the value of c: "))

# Calculate the expression
result = calculate_expression(a, b, c)

# Display output
print(f"\nThe result of a^5 + 2abc + b^3 + c^4 where a = {a}, b = {b}, and c = {c} is: {result}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named "Calculates the expression $a^5 + 2abc + b^3 + c^4$.py."

Explanation of code:

User input():

The program prompts the user to enter values for a, b, and c using the input() function. Each value is converted to a floating-point using float().

Variable Definition:

Variables like 'a', 'b', and 'c' store the values provided by the user. The 'result' variable stores the calculated value of the expression.

Expression Calculation:

The expression '$a^5 + 2abc + b^3 + c^4$' is calculated using process like 'a**5' for a^5, '$2 * a * b * c$' for 2abc, 'b**3' for b^3, and 'c**4' for c^4. The function 'calculate_expression()' performs this calculation and returns the result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to calculate the following expressions value when three values are entered as input in python. First, we must set up the variables for input number, create a function to perform expression calculation, using the exponentiation operator (**) to calculate the expressions, and display the results with the print() function.

III. $a^7 + 5a^3b^2c^6 + b^7$

Answer:

The given python code below calculates the expression $a^7 + 5a^3b^2c^6 + b^7$ by taking inputs for the variables a, b, and c.

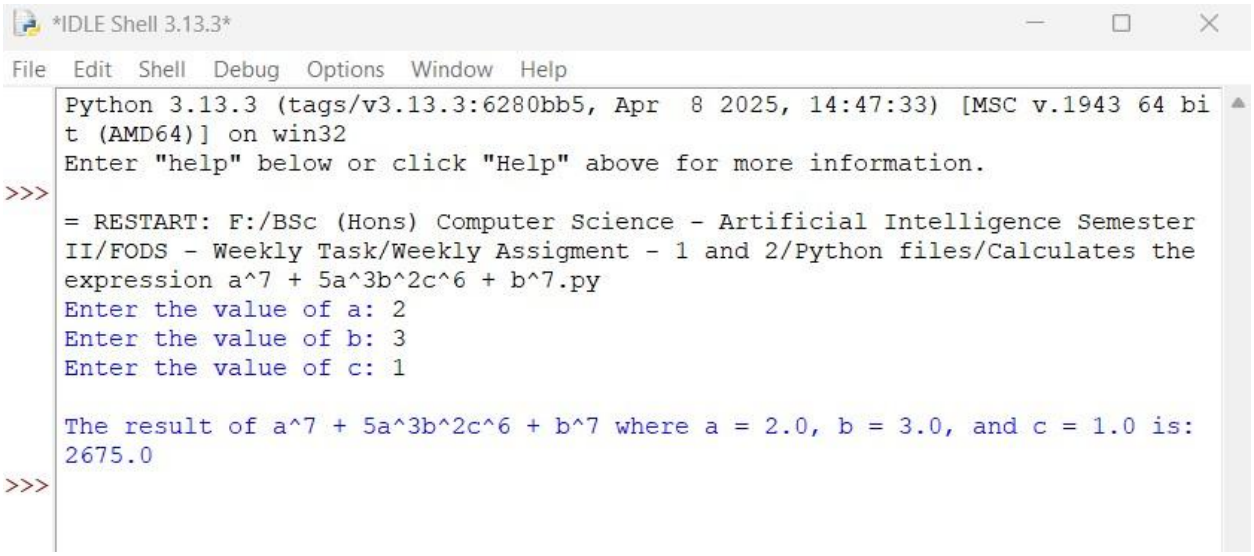
Following code for input:

Calculates the expression a^7 + 5a^3b^2c^6 + b^7.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS - We...

File Edit Format Run Options Window Help

```
'''
Program calculates the expression a^7 + 5a^3b^2c^6 + b^7 by taking inputs for variables a, b, and c.
'''
# Function to calculate the expression
def calculate_expression(a, b, c):
    return a**7 + 5*a**3*b**2*c**6 + b**7

# Input from user
a = float(input("Enter the value of a: "))
b = float(input("Enter the value of b: "))
c = float(input("Enter the value of c: "))

# Calculate the expression
result = calculate_expression(a, b, c)

# Display output of the program
print(f"\nThe result of a^7 + 5a^3b^2c^6 + b^7 where a = {a}, b = {b}, and c = {c} is: {result}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named "Calculates the expression $a^7 + 5a^3b^2c^6 + b^7$.py."

Explanation of code:

User input():

The program prompts the user to enter values for a, b, and c using the input() function. Each value is converted to a floating-point using float().

Variable Definition:

Variables like 'a', 'b', and 'c' store the values provided by the user. The 'result' variable stores the calculated value of the expression.

Expression Calculation:

The expression '$a^5 + 2abc + b^3 + c^4$' is calculated using process like 'a**5' for a^5, '$5 * a^3 * b^2 * c^6$' for $5a^3b^2c^6$, and 'b**7' for b^7. The function 'calculate_expression()' performs this calculation and returns the result.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to calculate the following expressions value when three values are entered as input in python. First, we must set up the variables for input number, create a function to perform expression calculation, using the exponentiation operator (**) to calculate the expressions, and display the results with the print() function.

6. Write a program to input the number of 5 subjects from the user, calculate the average, total, percentage and division of the students on the basis of specifications mentioned below.

I. If the percentage value is 80 or above, the user obtains distinction.

II. If the percentage is above 60, the user obtains first division.

III. If the percentage is above 50, the user obtains second division.

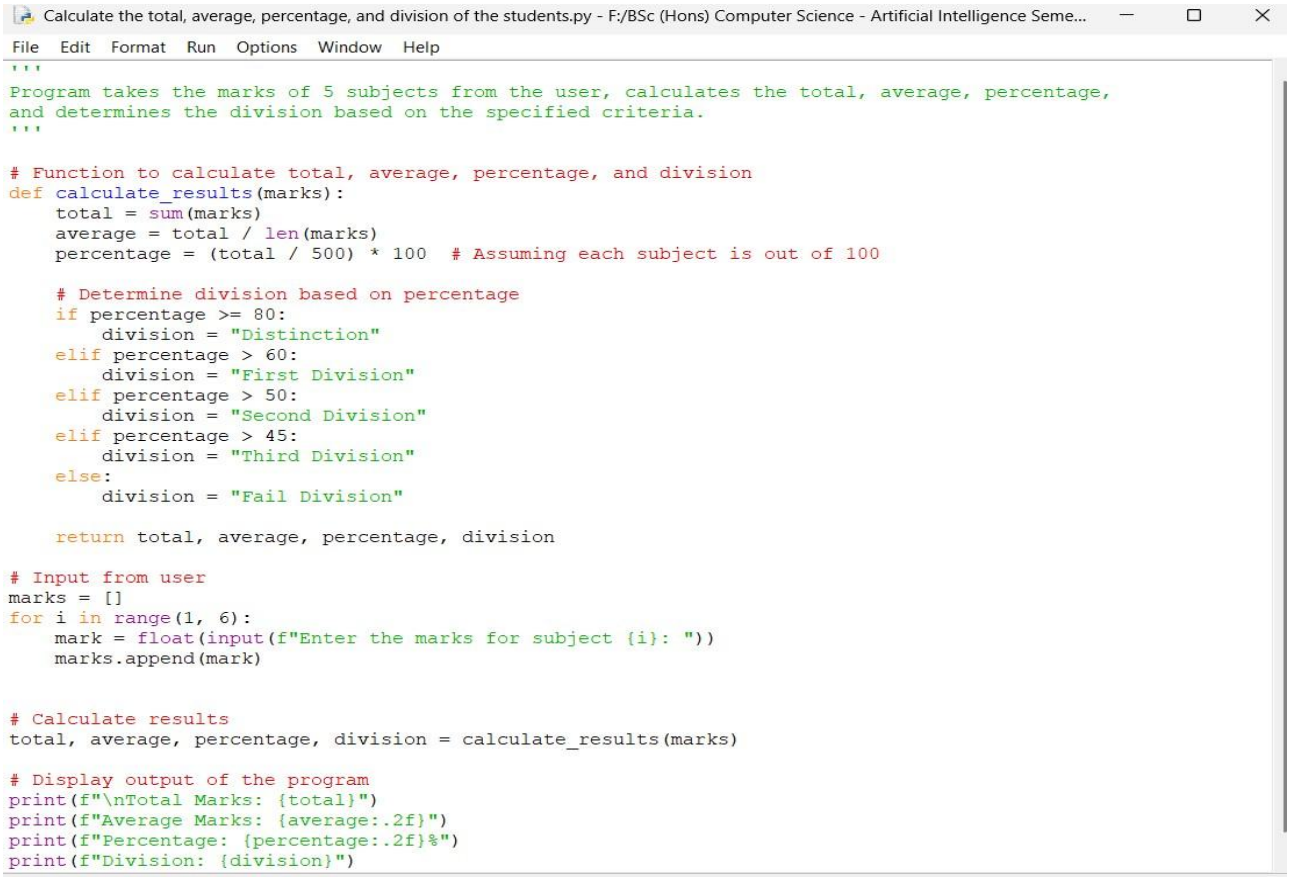
IV. If the percentage is above 45, the user obtains third division.

V. If the percentage is below 45, the user obtains fail division.

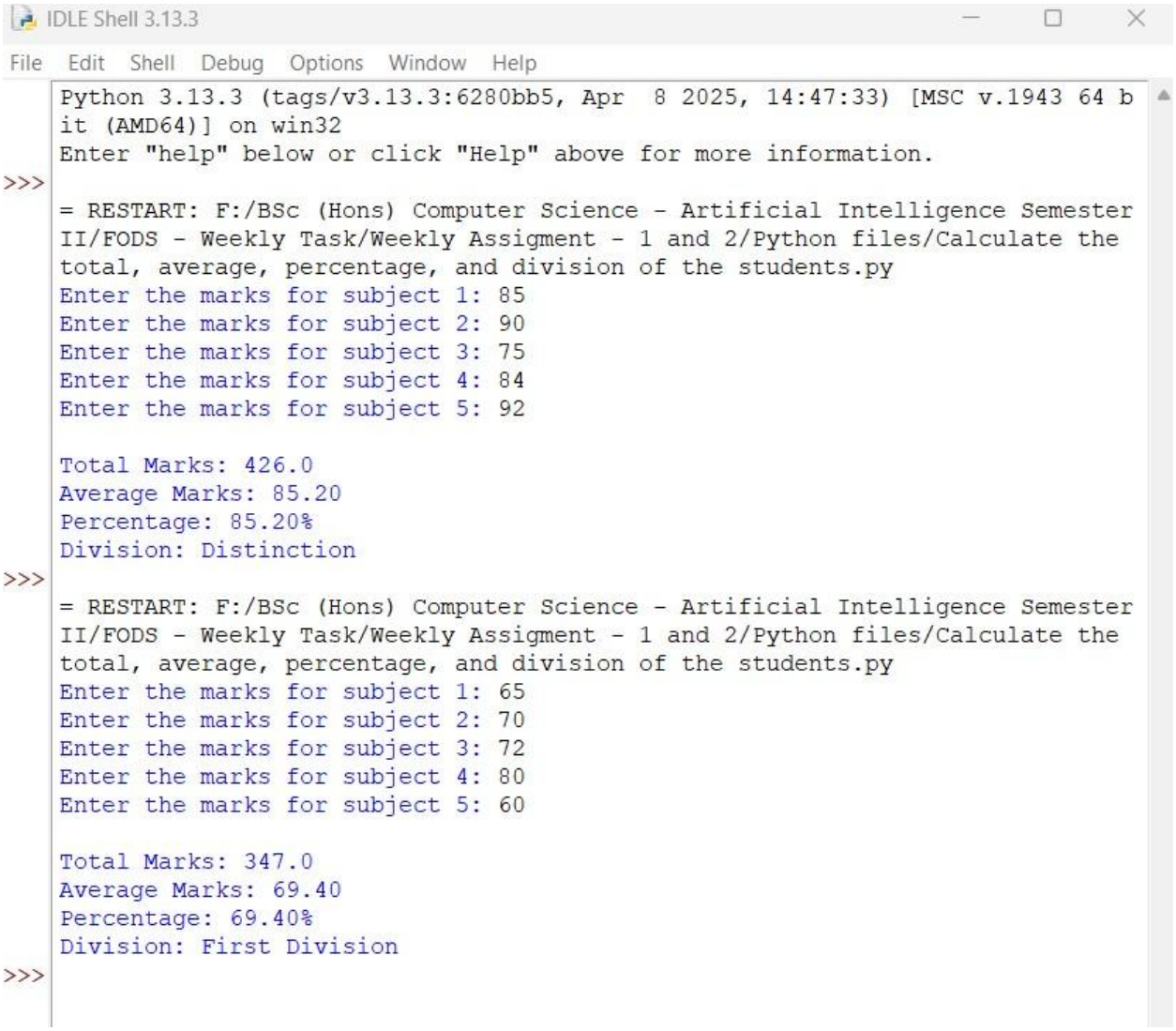
Answer:

The given python code below takes the marks entry of 5 subjects from the user, calculates the total, average, percentage, and determines the division based on the specified criteria for each student.

Following code for input:



Output obtained in execution:



Python Program File: Saved the above code in a file named “Calculate the total, average, percentage, and division of the students.py.”

Explanation of code:

User input():

The program prompts the user to enter the marks for five subjects using a loop. Each mark for subjects(1, 2, 3, 4, and 5) is converted to a floating-point using float().

Variable Definition:
The ‘marks’ list variable stores the marks for the five subjects provided by the user.

Result Calculation:

The ‘calculate_results’ function takes marks of each subject as input from the list.
Calculations:
Total: The sum of all marks.
Average: The average of the marks.
Percentage: The percentage calculated on the basis of total marks (assuming each subject is out of 100).
The function then determines the division based on the percentage found in the result using conditional statements.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program is an example of how to calculate the total marks, average, percentage, and the divisions of student by calculating their individual subjects in python. First, we must set up the variables for input marks, create conditional statement (if and elif) to make criteria for divisions, using for loop to make a list of marks achieved for various subjects, function to perform the resulting calculation and display the results with the print() function.

7. Write a program to display the prime number between 2 numbers input by the users. Also print the sum of all the prime numbers. [Hint: Prime numbers are the one which are either divisible by 1 or themselves. 3, 5, 7, 11, etc. are some of the examples.]

Answer:

The given python code below takes two numbers as input and displays all the prime numbers between the two given numbers. It also prints the sum of all the prime numbers found in the result.

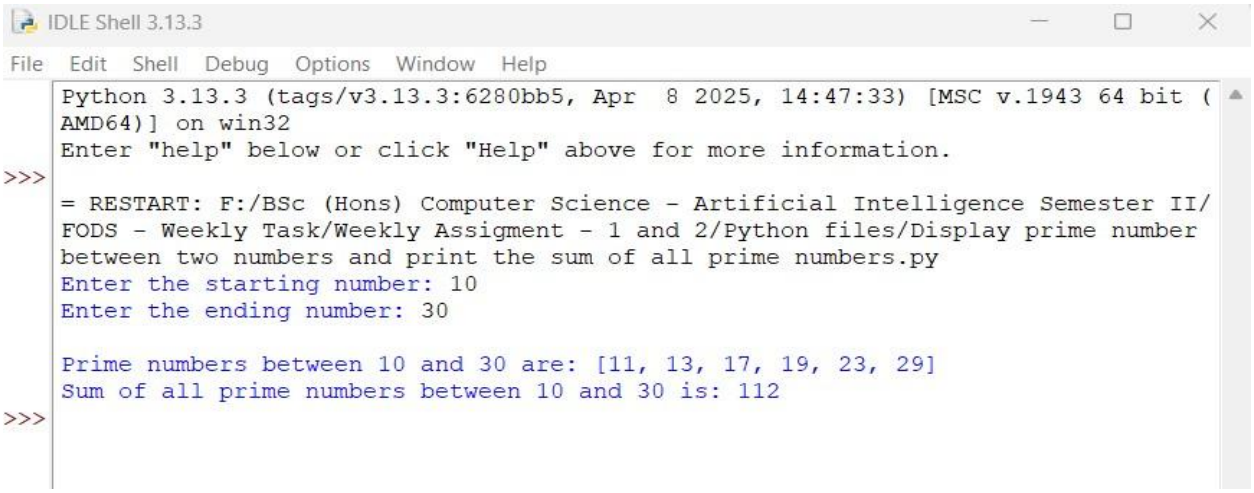
Following code for input:

Display the prime number between two numbers and print the sum of all prime numbers.py - F:/BSc (Hons) Computer Science - Artificial Inte...

File Edit Format Run Options Window Help

```
'''
Program takes two numbers as input from the user and displays all the prime numbers between those two numbers,
along with the sum of all the prime numbers found.
'''

# Function to check if a number is prime
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True


# Function to find prime numbers between two numbers
def find_primes_between(start, end):
    primes = []
    for num in range(start, end + 1):
        if is_prime(num):
            primes.append(num)
    return primes


# Input from user
start_num = int(input("Enter the starting number: "))
end_num = int(input("Enter the ending number: "))

# Find prime numbers between the two numbers
prime_numbers = find_primes_between(start_num, end_num)

# Calculate the sum of prime numbers
sum_of_primes = sum(prime_numbers)

# Display output of the program
print(f"\nPrime numbers between {start_num} and {end_num} are: {prime_numbers}")
print(f"Sum of all prime numbers between {start_num} and {end_num} is: {sum_of_primes}")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named “Display the prime number between two numbers and print the sum of all prime numbers.py.”

Explanation of code:

User input():

The program prompts the user to enter the starting and ending numbers using the input() function. Both inputs numbers are converted to integers using int().

Function to Check Prime:

The ‘is_prime(num) function checks if the given number is prime. It returns ‘False’ for numbers less than or equal to 1. If the number is divisible by any of these, it is prime.

Function to Find Primes:

The ‘find_prime_between (start, end)’ function generates a list of prime numbers between two input number. It iterates through each number for the given range and uses ‘is_prime’ function to check for prime numbers.

Finding and Calculating the Sum of Prime Numbers:

The program calls ‘find_prime_between’ function to get the list of prime numbers.

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program determines the prime number from given two numbers and displays the list of prime numbers found between those two numbers and calculates their sum.

8. Write a program which will find all such numbers that are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included). The numbers should be printed on the output screen. Also try the same program by replacing 2000 and 3200 by taking input for them from the users.

Answer:

The given python code below finds all the numbers between 2000 and 3200 that are divisible by 7 but not a multiple of 5. The program will also allow the user to input the range values instead of using fixed values.

Following code for input:

Find the numbers that are divisible by 7 but are not multiple of 5 between 2000 and 3200 and input custom range from the user.py...

File Edit Format Run Options Window Help

```
'''
Program finds all numbers between a given range that are divisible by 7 but not a multiple of 5.
'''

# Function to find numbers divisible by 7 but not a multiple of 5
def find_numbers(start, end):
    numbers = []
    for num in range(start, end + 1):
        if num % 7 == 0 and num % 5 != 0:
            numbers.append(num)
    return numbers

# Fixed range between 2000 and 3200
start_fixed = 2000
end_fixed = 3200


# Find numbers in the fixed range
numbers_fixed = find_numbers(start_fixed, end_fixed)
print(f"Numbers between {start_fixed} and {end_fixed} that are divisible by 7 but not a multiple of 5:")
print(numbers_fixed)


# Input from user for custom range
start_user = int(input("\nEnter the starting number: "))
end_user = int(input("Enter the ending number: "))


# Find numbers in the user-defined range
numbers_user = find_numbers(start_user, end_user)
print(f"\nNumbers between {start_user} and {end_user} that are divisible by 7 but not a multiple of 5:")
print(numbers_user)
```

Output obtained in execution:

IDLE Shell 3.13.3

File Edit Shell Debug Options Window Help

```
Python 3.13.3 (tags/v3.13.3:6280bb5, Apr  8 2025, 14:47:33) [MSC v.1943 64 bit (AMD64)]
on win32
Enter "help" below or click "Help" above for more information.
>>>
= RESTART: F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FODS -
Weekly Task/Weekly Assigment - 1 and 2/Python files/Find the numbers that are divisible
by 7 but are not multiple of 5 between 2000 and 3200 and input custom range from the us
er.py
Numbers between 2000 and 3200 that are divisible by 7 but not a multiple of 5:
[2002, 2009, 2016, 2023, 2037, 2044, 2051, 2058, 2072, 2079, 2086, 2093, 2107, 2114, 21
21, 2128, 2142, 2149, 2156, 2163, 2177, 2184, 2191, 2198, 2212, 2219, 2226, 2233, 2247,
2254, 2261, 2268, 2282, 2289, 2296, 2303, 2317, 2324, 2331, 2338, 2352, 2359, 2366, 237
3, 2387, 2394, 2401, 2408, 2422, 2429, 2436, 2443, 2457, 2464, 2471, 2478, 2492, 2499,
2506, 2513, 2527, 2534, 2541, 2548, 2562, 2569, 2576, 2583, 2597, 2604, 2611, 2618, 263
2, 2639, 2646, 2653, 2667, 2674, 2681, 2688, 2702, 2709, 2716, 2723, 2737, 2744, 2751,
2758, 2772, 2779, 2786, 2793, 2807, 2814, 2821, 2828, 2842, 2849, 2856, 2863, 2877, 288
4, 2891, 2898, 2912, 2919, 2926, 2933, 2947, 2954, 2961, 2968, 2982, 2989, 2996, 3003,
3017, 3024, 3031, 3038, 3052, 3059, 3066, 3073, 3087, 3094, 3101, 3108, 3122, 3129, 313
6, 3143, 3157, 3164, 3171, 3178, 3192, 3199]

Enter the starting number: 2000
Enter the ending number: 2500

Numbers between 2000 and 2500 that are divisible by 7 but not a multiple of 5:
[2002, 2009, 2016, 2023, 2037, 2044, 2051, 2058, 2072, 2079, 2086, 2093, 2107, 2114, 21
21, 2128, 2142, 2149, 2156, 2163, 2177, 2184, 2191, 2198, 2212, 2219, 2226, 2233, 2247,
2254, 2261, 2268, 2282, 2289, 2296, 2303, 2317, 2324, 2331, 2338, 2352, 2359, 2366, 237
3, 2387, 2394, 2401, 2408, 2422, 2429, 2436, 2443, 2457, 2464, 2471, 2478, 2492, 2499]
>>>
```

Python Program File: Saved the above code in a file named “Find the numbers that are divisible by 7 but not a multiple of 5 between 2000 and 3200 and input custom range from the users.py.”

Explanation of code:

User input():

The program prompts the user to enter their own starting and ending numbers. It then, converts the input numbers to integers and calls the ‘find_numbers’ function with these values.

Function to Find Numbers:

The ‘find_numbers(start, end)’ function iterates through the range from ‘start’ to ‘end’. It checks each number to see if it is divisible by 7 (num % 7 == 0) and not a multiple of 5 (num % 5 != 0). If both conditions are met, then the number is added to the list of numbers.

Fixed Range:

The program first uses a fixed range from 2000 to 3000 to find and print the numbers that meet the given criteria.

User Input for Custom Range:

The program prompts the user to enter their own starting and ending numbers. Then, it converts the input to integers and calls the ‘find_numbers’ function with these values.

Output:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program displays the list of numbers that are divisible by 7 but not a multiple of 5 for both the fixed range (2000 - 3200)and user defined ranges.

9. Write a program to find the factorial of any number taken as an input from the user. Try to validate the user input whether it is a number or not and then only perform the operation. In case of other than number as an input, display an error as “Not a number.”. [Hint: few available functions to identify the input is a number or not are ‘isdigit(), isnumeric(), etc.]

Answer:

The given python code below that calculates the factorial of a number provided by the user. The program validating the user input to ensure if is a positive number and handles error. Displays error if it is non-positive number.

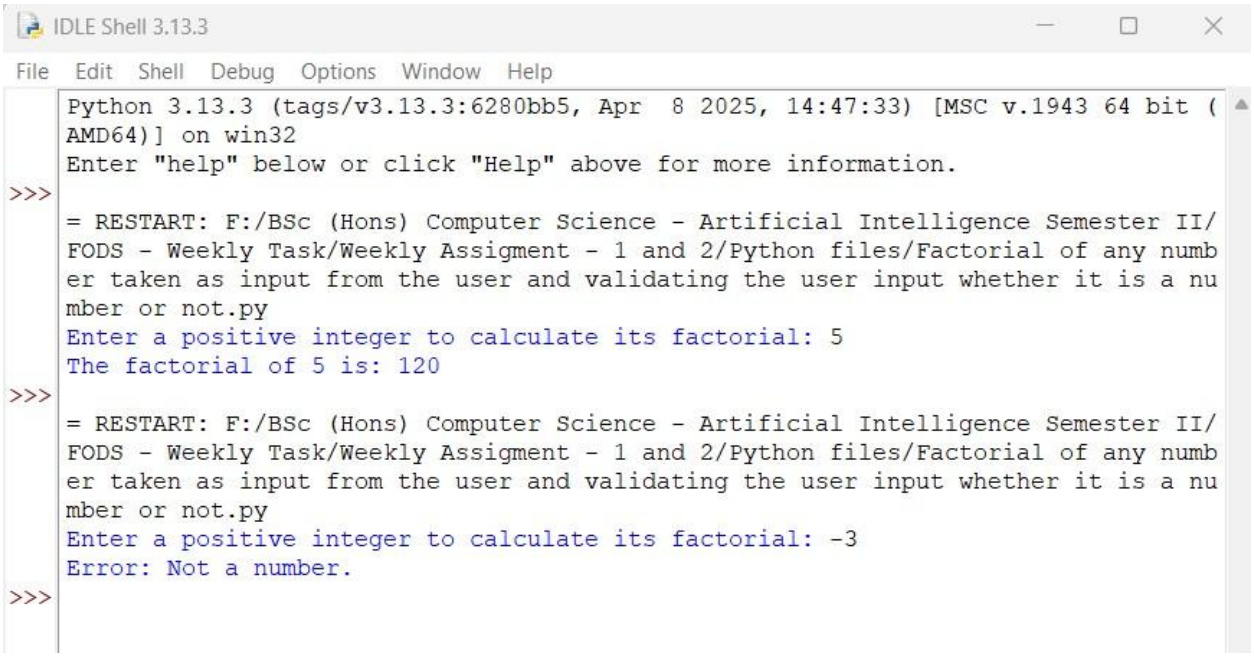
Following code for input:

Factorial of any number taken as input from the user and validating the user input whether i...

File Edit Format Run Options Window Help

```
'''
Program calculates the factorial of a number taken as input from the user.
It validates the input to ensure it is a positive integer.
Check for error in the program.
'''

# Function to calculate factorial
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        result = 1
        for i in range(2, n + 1):
            result *= i
        return result


# Input from user
user_input = input("Enter a positive integer to calculate its factorial: ")


# Validate input
if user_input.isdigit():
    number = int(user_input)
    if number < 0:
        print("Error: Please enter a positive integer.")
    else:
        # Calculate factorial
        fact = factorial(number)
        print(f"The factorial of {number} is: {fact}")
else:
    print("Error: Not a number.")
```

Output obtained in execution:



Python Program File: Saved the above code in a file named “Factorial of any number taken as input from the user and validating the user input whether it is a number or not.py.”

Explanation of code:

User input():

The program prompts the user to enter a positive number using input() function.

Function to Calculate Factorial:

The factorial(n) function calculates the factorial of a given non-negative integer n. Program uses a loop to multiply all the integers from the range of 2 to n to calculate the factorial. The function returns 1 for the cases when n is 0 or 1, as 0! = 1! =1.

Input Validation:

The program checks if the input value is a digit using the ‘isdigit()’ method. This method returns ‘True’ if the string consists only of digits. If the input value is valid, it converts the string to an integer. It also checks if the number is negative and displays a message if it is or not.

Calculating and Displaying the Factorial:

If the input is valid as non-negative, then the program calls ‘factorial’ function and displays the result.

Error Handling:

If the input number is not a number, the program displays an error message: “Error: Not a number.”

Output of the program:

Finally, the program uses the print() function to show the result.

Conclusion:

The given program displays the factorial of a given number entered by the user. It includes a feature like error handling to check if positive number is entered or not.

10. Write a program to find the sum of odd and even numbers input from the user. The program should take an input continuously until the user wants to exit the program. The program should be menu driven where the user should be asked whether they want to continue with the program or not.

Answer:

The given python code below that takes input from the user to find the sum of odd and even numbers. The program is menu-driven based, allowing the users to decide whether to continue or exit the program.

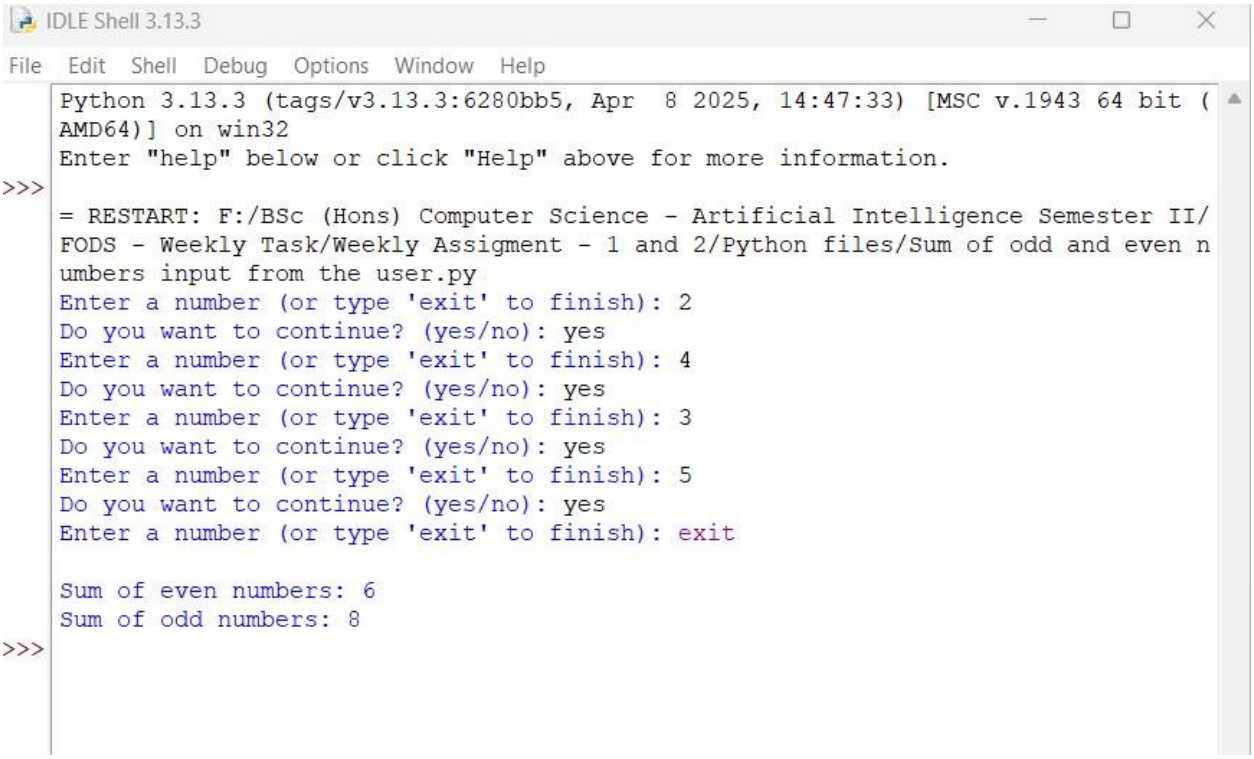
Following code for input:

Sum of odd and even numbers input from the user.py - F:/BSc (Hons) Computer Science - Artificial Intelligence Semester II/FOD...

File Edit Format Run Options Window Help

```
'''
Program continuously takes input from the user to find the sum of odd and even numbers.
Program is menu-driven, allowing the user to continue or exit the program.
'''

# Function to calculate the sum of odd and even numbers
def calculate_sums(numbers):
    sum_even = sum(num for num in numbers if num % 2 == 0)
    sum_odd = sum(num for num in numbers if num % 2 != 0)
    return sum_even, sum_odd

# Main program loop
def main():
    numbers = []

    while True:
        user_input = input("Enter a number (or type 'exit' to finish): ")

        if user_input.lower() == 'exit':
            break

        # Validate input
        if user_input.isdigit() or (user_input.startswith('-') and user_input[1:].isdigit()):
            numbers.append(int(user_input))
        else:
            print("Error: Please enter a valid integer.")
            continue

        # Ask if the user wants to continue
        continue_input = input("Do you want to continue? (yes/no): ").strip().lower()
        if continue_input != 'yes':
            break

    # Calculate sums
    sum_even, sum_odd = calculate_sums(numbers)

    # Display results
    print(f"\nSum of even numbers: {sum_even}")
    print(f"Sum of odd numbers: {sum_odd}")

# Call the main function to run the program
main()
```

Output obtained in execution:



Python Program File: Saved the above code in a file named “Sum of odd and even numbers input from the user.py.”

Explanation of code:

Function to Calculate Sums:

The ‘calculate sums(numbers)’ function takes a list of numbers as input from the user. It calculates the sum of even numbers using a generator expression and sum() function.

Main Program Loop:

The main() function contains a loop that runs indefinitely until the user decides to exit it. The user is prompted to enter a number or type ‘exit’ to finish the input. If the user types exit, the loop breaks, and the program calculates the sums as the next step.

Input Validation:

The program checks if the given input number is a valid integer (including negative numbers). If the input is valid, it is converted to an integer and added to the ‘numbers’ list. If the input is invalid, then a error message is displayed, and the loop continues.

Continue Prompt:

After each valid input is entered, the user is asked if they want to continue. If the user types anything other than ‘yes’, the loop breaks.

Output of the Program:

If we exit the loop, the program calculates the sum of odd and even numbers and displays the results. Finally, the program uses the print() function to show the result.

Conclusion:

The given python program displays sum of even numbers and sum of odd numbers. It includes a feature like menu-driven interface which allows user to continue or exit the program.

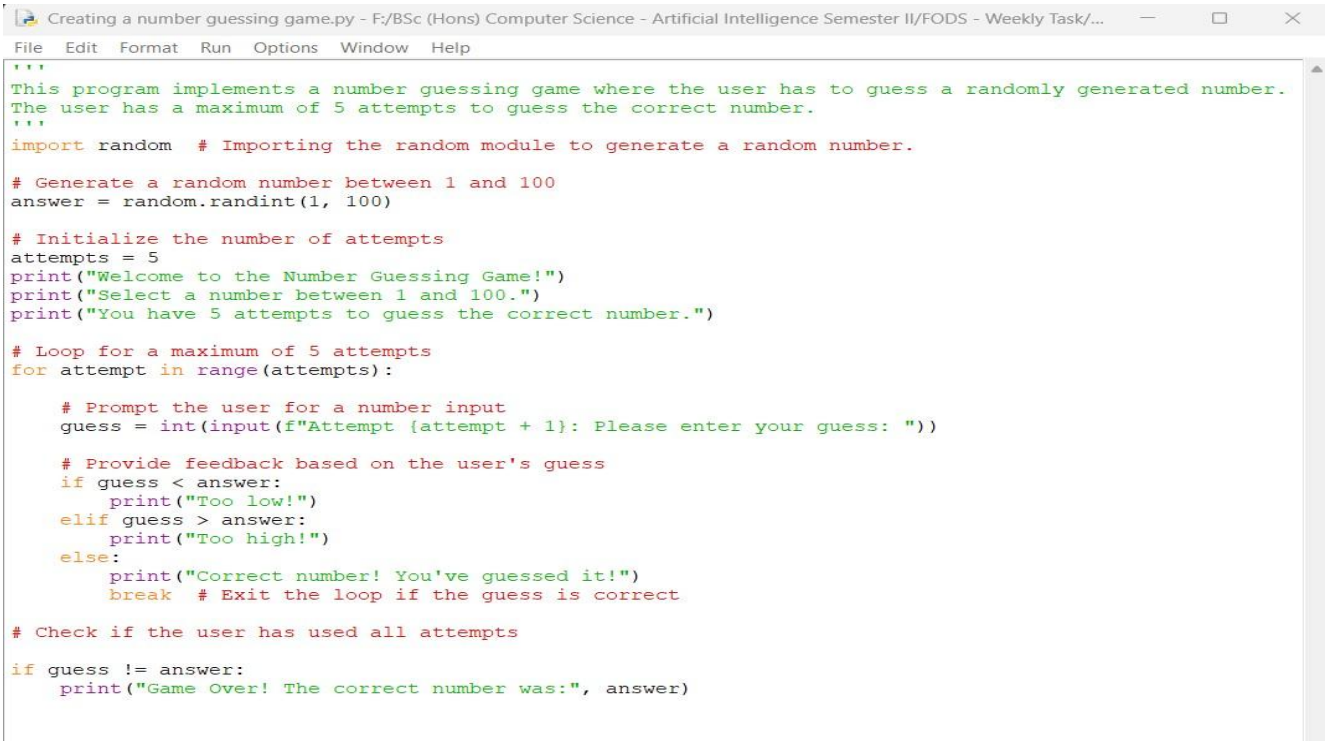
11. Write a program to create a number guessing game for the user. The program should ask the user to input a number. The program specifications are as mentioned below.

I. The program should generate a random number for the answer.
II. The program should prompt the user for a number input.
III. The program should provide the feedback to the user after each guess (e.g. “Too high”, “Too low” or “Correct number”).
IV. The program should check the user input for 5 times and allow the users to guess for at most 5 times if their input don’t match the answer number.
V. If the user is not able to guess the answer within 5 times, the program should display “Game Over” message and exit.

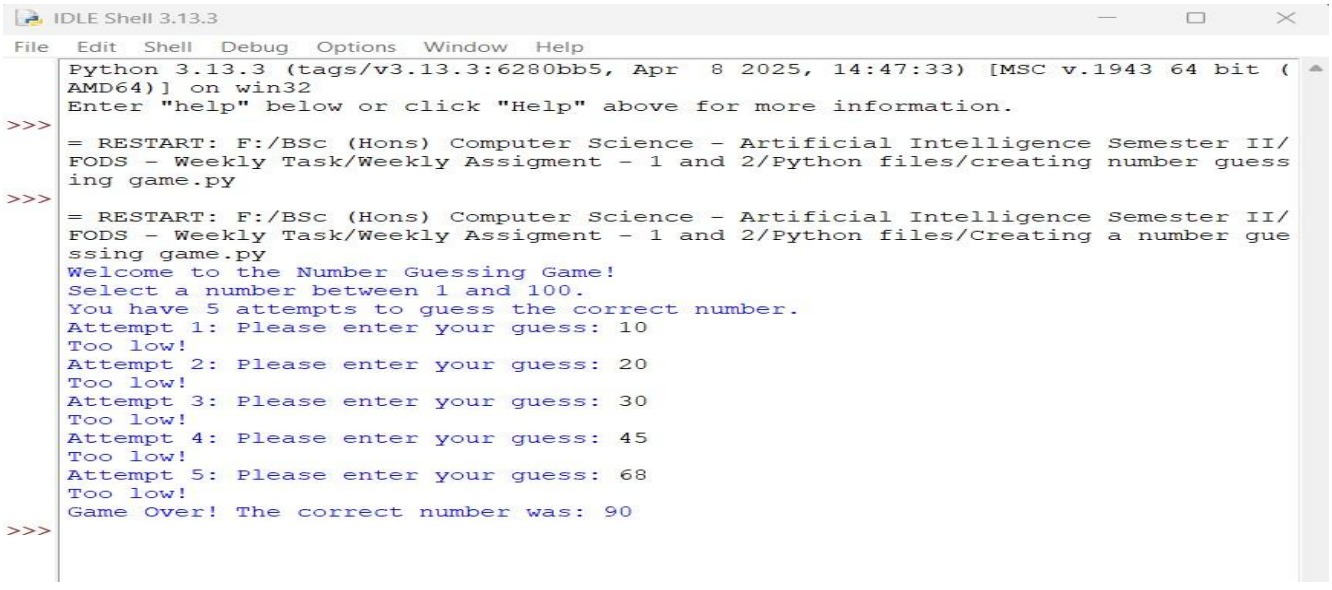
Answer:

The given python code below is used to create a guessing name game that will be asked by the user for number input. The user has a chance to guess the name in 5 attempts.

Following code for input:



Output obtained in execution:



Python Program File: Saved the above code in a file named “Creating a number guessing game.py.”

Explanation of code :

Importing the Random Built-in Module:

The program imports by bringing in the random built-in module, which is essential for generating a random number.

Random Number Generation:

It then creates a random number between 1 and 100 using random.randint(1, 100) and saves it in a variable called answer.

Initialization for Number of Attempts:

Next, the variable attempts are set to 5, indicating the maximum number of guesses the player can make.

Game Introduction:

The program welcomes the user with a friendly message and lays out the instructions for how to play.

Loop for User Guesses:

A for loop is used to give the user up to 5 chances to guess the number. This loop runs through the range of attempts.

User Input:

Within the loop, the program asks the user to enter their guess and converts that input into an integer.

Feedback Mechanism in the program:

The program then checks how the user's guess and compares it against the generated number:

If the guess is lower than the answer, it responds with "Too low!".

If the guess is higher than the answer, it responds with "Too high!".

If the guess is spot on, it exclaims "Correct number!" and exits the loop.

Game Over Message and output of the program:

Once the loop is finished, if the user has not guessed the right number, the program delivers a "Game Over" message along with the correct answer.

Conclusion:

The given program is an example of number guessing game built in Python. First, we must use loops, conditional statements, and gather user input to create an interactive user experience.