Experiment No.2#c

Frason Francis : 201903020 : 25

Aim: To study advanced Data types and functions in Python.

1.Python program for finding a power of a given number using Recursive Method
2.Python Program for printing Fibonacci Number using Recursive method.

Theory:

There are four different types in Python:

1. int(plain integers): this one is pretty standard -plain integers are just positive or negative whole numbers.
2. long (long integers): long integers are integers of infinite size. They look like plain integers except they're followed by letter "L".
3. float (floating point real values): floats represent real numbers, but are written with decimal points(for scientific notation) to divide the whole number into fractional parts.
4. complex(complex numbers): Represented by the formula a+bj where a and b are floats,and j is the square root of -1 (the result of which is an imaginary number). Complex numbers are used sparingly in Python.
5. A tuple is a collection type data structure which is immutable by design and holds a sequence of heterogeneous elements.
6. Tuples store a fixed set of elements and don't allow changes whereas the list has the provision to update its content.
7. Python Set Data Structure: Python Set represents a group of unique elements. If you wish to describe a group of unique items into a single entity, then you can go with Python Set. The Set doesn't allow duplicate elements. It doesn't preserve the insertion order. We can store the heterogeneous elements in a Set. Set objects are mutable.

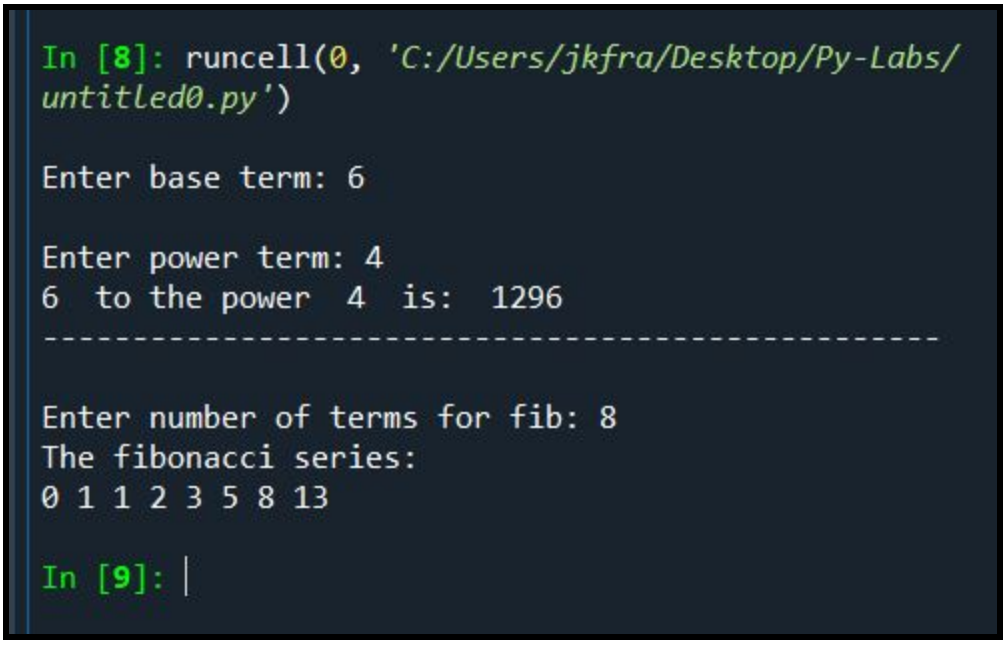
Algorithms:

1.
2. Begin
3. Take x and y as input from the user
    a. X = base term
    b. Y = power term
    c. Result = pow(x,y)
4. Define pow function(x,y)
    a. If y == 1
        i. Return x
    b. Else
        i. Return pow(x,y-1)*x
5. Take the input num from the user to define no. of fib
6. Define fibseries(num)
    a. If num <= 1
        i. Return num
    b. Else
        i. return fibSeries(num-1) + fibSeries(num-2)
7. Exit

Codes:

```
def pow(x, y):
    if y == 1:
        return x
    else:
        return pow(x, y-1) * x

def fibSeries(num):
    if num <= 1:
        return num
    else:
        return fibSeries(num-1) + fibSeries(num-2)

# main code
if __name__ == '__main__':

    x = int(input("Enter base term: "))
    y = int(input("Enter power term: "))
    result = pow(x, y)
    print(x," to the power ", y, " is: ", result)


print("-"*50)

# take input
num = int(input('Enter number of terms for fib: '))

# print fibonacci series
if num <= 0:
        print('Please enter a positive integer')
else:
    print('The fibonacci series:')
    for i in range(num):
        print(fibSeries(i), end=' ')
```

Output:



Conclusion:

In this experiment we have successfully implemented recursion and performed the given experiment.