Python Programming

Practice – 1

Overview

This practice reinforces the basics of Python Programming.

- *It helps to understand the different modes in which Python works.*
- *The basic data types of Python, the numbers and strings.*
- *Arithmetic and String operators.*
- *IO operation with Python programs.*

We shall put these elements together to create simple basic Python programs.

Hands On

1. Start **Python** in ***Interactive Mode*** and understand the following:.

    a) The different **keywords** of Python
    b) The basic **arithmetic operators** of Python
    c) The different **number types** supported by Python
    d) The use of **type( )** function for RTTI
    e) The Python **special variable _** (underscore)
    f) Different representations of **Python strings**.
    g) The different **string operators**
    h) Strings Features like : String slicing, Immutable property of string, Usage of negative subscripts with stings.

2. Write a program to find the area and circumference of a circle.

```
Area = PI * Radius^2
Circumference = 2.0 * PI * Radius
```

   NOTE : Import the **math** module (import math) and use the **PI** constant as **math.pi** in this program.

3. Write a program to find the temperature is in Fahrenheit, given the temperature in Celsius equivalent and vice-versa.

```
*Celsius = ( 5.0 / 9.0 ) * ( Fahrenheit - 32.0 )
*Fahrenheit = (( 9.0 / 5.0 ) * Celsius ) + 32.0
```

Practice – 2

Overview

This practice session enables us to understand the different control structures of Python programming language like:

- *Selection Statement*
- *Iterative Statements*
- *Branch Statements*

We make use of control structures to write Python programs which make decisions and repetitions.

Hands On

1. Write a program to find the given number is Even or Odd.

2. Accept a year from the user and find whether it is a leap year or not.

   HINT : A leap year is divisible by 4 and not by 100, or is divisible by 400
   NOTE: Try the above program with nested selection statements and an selection statement with a compound condition.

3. Accept three positive integers from the user representing the three sides of a triangle. Determine whether they form a valid triangle or not.

   HINT : In a triangle, the sum of any two sides must always be greater than the third side.

4. Write a program to generate a multiplication table for a given integer.

   [a] Using **while** repetition construct
   [b] Using **for** repetition construct

5. Write a program to find the factorial of any positive integer.

6. Write a program to reverse a given integer.

7. Write a program to check the given number is Prime or not.

8. Write a program to generate all Prime numbers between 1 to 999

   NOTE : Make use of else clause with the iterative construct while generating Prime numbers.

9. Write a program to print all the reciprocals between -5 to +5

10. Print every number from 1 to 20 in base 8 and base 16 along with its decimal equivalent in tabular format.

11. Write a program to find the roots of a quadratic equation.

12. Write a program to get a count of vowels, consonants and other characters in a given string.

13. Check if the given string is a Palindrome or not.

14. Enhance the multiplication table generation (Question 4) program such that it helps us in repeatedly generating the table until the user wants.

15. Write a program to generate all **Amstrong** numbers between 1 to 999

    NOTE : Amstrong numbers are those numbers where the sum of the cubes of the individual digits equals the number itself.

16. Write a program to implement the number guessing game.

Python Programming

Practice – 3

Overview

This practice reinforces us with the understanding of compound data types in Python.

- *Manipulation of List and Tuples.*
- *Using Sets and performing Set operations like Union, Intersection etc.*
- *Creation & Manipulation of Dictionaries.*

We shall put these elements together to create simple basic Python programs.

Hands On

1. Write a menu driven program with the following option to implement stack using the list data structure.

```
S T A C K   M E N U

[a] Push.
[b] Pop.
[c] Display
[d] Quit
```

2. Write a menu driven program with the following option to implement queue using the list data structure.

```
Q U E U E     M E N U

[a] Insert Into the Q.
[b] Delete From the Q.
[c] Display Q
[d] Quit
```

3. Accept a **`POSTFIX`** expression from the user and compute it. Use list as the data structure for stacking purpose.

4. Accept data into the N elements of an list and perform the following:

   [a] Linear Search
   [b] Binary Search

Compiled By : *Mohammed Mukthar Ahmed*

5. Initialize a list with the following values, (each represents the maximum number of days in a month). Accept month and year from the user and display the maximum number of days in that month.

   31 28 31 30 31 30 31 31 30 31 30 31

   NOTE : Care has to be taken in case of leap year (Feb will be 29 days)

6. Write a program to accept the marks of students for two subjects. The average of the marks secured by the student is calculated and stored in an list

   [a] The maximum in each subject is 100. Perform necessary validation.
   [b] Find the number of students whose average lies in the range
   < 40
   >= 40 to < 60
   >= 60 to < 80
   >= 80 to < 100
   ==100
   [c] Display the averages in ascending / descending order as per user's wish
   [d] Display the highest and lowest average marks.

7. Start Python in Interactive Mode and perform the following:

   [a] The creation of TUPLE data structure
   [b] Manipulation of TUPLE
   [c] The immutable property of TUPLE
   [d] Tuple packing and unpacking

8. Write a program to illustrate the **SET** data structure along with the following operations:

   [a] Membership test
   [b] Set Operations : Union, Intersection & Difference

9. Write a program to find the Roman equivalent for a given single digit number using dictionary data structure.

10. A dictionary stores the Employee ID and other employee details like Name, Gender, Business Unit and Basic Salary.
    Write a program to find the Net Pay for the given Employee ID

```
Net Pay = Basic Salary + HRA + DA – PF


HRA is 20% of the Basic Salary
```

```
DA is 12% of the Basic Salary
PF is 8% of the Basic Salary
```

11. A dictionary holds the Country Name and it President.
    Write a program to have a menu with the following options.

    [a] Given the country name the President is obtained.
        Ensure that country name which is not part of the dictionary is taken care.
    [b] Display all the countries with their respective Presidents.
    [c] Remove the key-value pair for a given key.

    NOTE : Use only single words to represents countries and presidents.

12. The ‘**os**’ module has a built-in dictionary by name ‘**environ**’.
    Write a program to simulate the ‘**set**’ command.

Session – 3b

1. Accept data into N elements of a list from the user.

   [a] Using the **filter( )** function display all even and odd numbers separately from a list of number.
   [b] Using the **map( )** function and double the contents for the given list.

2. Using the List Comprehension method perform the following:

   [a] Convert the Celsius values info Fahrenheit for the given list
   Celsius = [39.2, 36.5, 37.3, 37.8]

   ```
   *Celsius = ( 5.0 / 9.0 ) * ( Fahrenheit - 32.0 )
   *Fahrenheit = (( 9.0 / 5.0 ) * Celsius ) + 32.0
   ```

   [b] Identify Pythagorean triples for all the number in the range of 1 to 20.

3. A list of 10 elements hold both positive and negative numbers (single digit) Wirte a program to generate an Histogram based on the contents of the list as shown below:

   ```
   List
   a[ 0 ] = 5;         + + + + +
   a[ 1 ] = -3;        - - -
   a[ 2 ] = -6;        - - - - - -
   a[ 3 ] = 8;         + + + + + + + +
   ```

4. Write a program to get the following output

   ```
   1
   1 2
   1 2 3
   1 2 3 4
   1 2 3 4 5
   ```

5. Write a program to illustrate that logical operators are short-circuited.

Python Programming

Practice – 4

Overview

This practice reinforces us to understand the need and creation of user-defined functions or custom functions in Python.

- *Define and invoke functions.*
- *Passing parameters and returning values from functions.*
- *Default argument values and Keyword arguments.*
- *Building functions with arbitrary number of arguments.*
- *Building short anonymous functions.*

We shall put these features and build user-defined Python functions.

Hands On

1. Write user-defined function for the following

   To display a message

   ```
   "Welcome to BANGALORE"
   "Have a nice day!"
   ```

2. Write a function which finds the cubes of numbers from 1 to 5.

3. Write a function to do the following tasks

   [a] Check if the year passed as an argument is a leap year or not.
   [b] Check if the integer passed as an argument is Prime or not.
   Return a Boolean value.

4. Write a function to determine the roots of a quadratic equation.

5. Write a function which returns a tuple of the indices of the two smallest values in list.

6. Write a function to check the given character is:

   [a] A upper case letter,
   [b] A lower case letter,
   [c] A digit or
   [d] A special symbol

7. Write a function with default argument value to compute simple interest. The default rate of interest is 10% otherwise the user specifies it.

8. Write a function with default arguments to print the specified character, the specified number of times.

   The default character is “ * “ and the default number is 40

9. Write a function which takes two argument, the first being the temperature and the second being the character to indicate whether the temperature is in Fahrenheit (F) or Celsius (C).

   If the temperature is in Fahrenheit, the function should calculate and return the Celsius equivalent and vice-versa.

   ```
   *Celsius = ( 5.0 / 9.0 ) * ( Fahrenheit – 32.0 )
   *Fahrenheit = (( 9.0 / 5.0 ) * Celsius ) + 32.0
   ```

   Use the keyword argument mechanism to implement the function.

10. Write a function which takes three argument, the first being the title, the second being the name and the third being the message.

    ```
    Message(title, name, msg)
    ```

    Use the keyword argument mechanism and display the information with the following print function:

    ```
    print( “%s %s \n %s”  % (title, name, msg))
    ```

11. Write a recursive function to find the factorial of a number.

    [a] Check if the function is working properly or not.
    [b] Find the Binomial co-efficient `nCr = n! / (n – r )! * r!`

12. Write a recursive function to find the GCD of two positive numbers.

    [a] Using the above function find the `LCM = ( m * n ) / GCD(m, n)`

13. Write a recursive function to get the Nth Fibonacci number.

14. Write a recursive function to solve the problem of **Tower of Hanoi**.

15. Using the variable argument list mechanism write a function which returns the sum of all the integers passed to it as arguments.

16. Write Python program to illustrate the different type of scopes for a variable and the usage of global statement.

17. Modify the Tower of Hanoi recursive function such that function also gives a count of the number of disk moves. The count is held in a global variable.

18. Demonstrate the use of anonymous function using lambda.

    [a] To find the area of a right angle triangle
    [b] To find the volume of a box.
    [c] To convert inches to centimeters.

19. Provide document strings for all the above user-defined functions with text indicating their usage.

Python Programming

Practice – 5

Overview

This practice reinforces us to understand the need for creating modules in Python.

- *Creating and importing modules.*
- *Understand where Python searches for module files.*
- *Creation and usage of Python compiled files.*
- *Using partial components of a module.*
- *Peep through and get help for a module.*

We shall put these elements of modules in creating and using Python modules.

Hands On

1. Create a module by name UDF, such that it holds all the user defined functions (Refer Q1. and Q6 of the previous session)

   Write program which uses the UDF module and demonstrates the use of the user defined functions.

2. Create a module by name DefaultArgs, such that it holds all the user defined Functions which have default argument (Refer Q7. and Q8 of the previous session)

   Write program which uses the DefaultArgs module and demonstrates the use of the user defined functions with default arguments.

3. Create a module by name KeyArgs, such that it holds all the user defined Functions which have keyword argument mechanism (Refer Q9. and Q10 of the previous session)

   Write program which uses the KeyArgs module and demonstrates the use of the user defined functions with keyword arguments.

4. Create a module by name Recursion, such that it holds all the user defined Recursive Functions (Refer Q11. and Q14 of the previous session)

   Write program which uses the Recursion module and demonstrates the use of recursive functions.

   Give local name to the Recursion module function and execute them.

5. Import the UDF module using the **from** statement

    [a] with specific user defined function
    [b] with all user-defined function using “ * “

6. If the modules are not being identified, we need to set the module search path. Are your modules being executed if they are placed in a different directory, if no perform appropriate action such that they start working?

7. Using the standard module “sys” change the primary and second prompts of the Python environment.

8. Observe the different objects of the __builtins__ module.

9. Create a package, such that we are in a position to use all our function created in the previous session as follows:

    MyFunctions
    MyFunctions.UDF
    MyFunctions.DefaultArgs
    MyFunctions.KeyArgs
    MyFunctions.Recursion

Python Programming

Practice – 6

Overview

This practice reinforces us to understand the need for fancier output besides file handling techniques.

- *Fancier formatted output.*
- *File Handling.*
- *Random Accessing.*
- *Command line arguments.*
- *Redirection.*

We shall put these elements in creating Python programs in this practice session.

Hands On

1. Write program to convert the given number to a string
   .
   [a] Using the **str( )** function
   [b] Using the **repr( )** function

2. Using the string justification methods generate a table for the square and cubes of the first ten numbers.

   [a] Use different justification like, left, right and center and observe the output

3. Write a program which writes the given phrases to a text file by name "**PHRASES.TXT**" in the current directory.

4. Write a program to read the contents of a specified text file line by line.

5. Write program to write the following data into a file **CUSTMER.DAT**

   [a] The customer details are as follows :
   Customer ID, Name and Age
   The details are delimited by (" ~ ") tilde sign.

   [b] The customer details are accepted from the user and written into the data file until the user has no more records to be written.

   [c] The same file could be used to add additional records later. Thus ensure an appropriate file opening mode.

Compiled By : *Mohammed Mukthar Ahmed*

6. Write a program to illustrate random access mechanism in context to files

7. Write a program to illustrate command line arguments.

8. Write program to simulate the following using command line arguments.

   [a] The **cat** command of UNIX
   [b] The **nl** command of UNIX
   [c] The **head** and **tail** command of UNIX

9. The **os** module environ directory hold all the key-value pairs. Write a program to simulate the **set** or the **env** command.

10. Set the standard output to a file by name "**OUTPUT**" and perform output redirection.

Python Programming

Practice – 7

Overview

This practice reinforces us to understand the need for exceptional handling in Python.

- *Understanding what are Exceptions.*
- *Exception Handling using **try ... except**.*
- *Handling multiple exceptions.*
- *Handling indirect exceptions.*
- *Raising an exception.*
- *Defining Clean-up action.*

We shall put these elements in creating Python programs which handle exceptions in this practice session.

Hands On

1. Execute the programs (Q1, Q2, or Q3 of Practice – 2). Enter some text when a number is asked. Record your observation.

2. Use appropriate exception handling method such that the user enters only the required data.

3. Execute the program (Q4 of Practice-6) and record your observation for the following:

   [a] **The specified file does not exist**
   [b] **You have no read permission**
   [c] **It is not an ordinary text file**

   Use multiple exception handling mechanism and overcome the above expected errors.

4. Write a program to illustrate the optional **else** clause of the **try…expect** statement

5. Write a program to illustrate Indirect Exception.

6. Write a program which accepts the day, month and year and stores them as the first three elements of a list.

   [a] Raise an “Invalid Month” exception when the month is not between 1 to 12
   [b] Raise an “Invalid Day” exception when the day is not between 1 to 30.
   [c] Take care of leap year.

7. Write a program to illustrate the **finally** clause of the **try…expect** statement.

8. Modify all other programs which you have created in your previous practice sessions to handle expected exceptions.

Python Programming

Practice – 8

Overview

This practice reinforces us to understand the Object-Oriented features of Python.

- *Define Class and Instantiate Objects.*
- *Access Attributes and Methods.*
- *Class Variables.*
- *Understand Encapsulation, Information Hiding & Inheritance.*
- *Perform instance test with objects.*
- *Perform subclass test in Inheritance.*

We shall put these elements in creating Python programs in this practice session.

Hands On

1. Create a class by name “**Person**” who’s object hold the following information

   The Name, Gender and Age of the person
   Gender is either “M” or “F”.
   Age should be between 1 and 99 only

   Write appropriate methods to initialize the object, display the objects data.

2. Create a class by name “**Date**” who’s objects hold the day, month and the year.

   Write appropriate methods to perform the following:
   [a] Initialize the date object
   [b] Display the date in dd/mm/yyyy i.e. **British Format**
   [c] Display the date in mm/dd/yyyy i.e. **American Format**
   [d] Return only the day, the month or the year
   [e] Display the date in dd-mon-yyyy format. Where mon represents Jan, Feb..
   [d] Display the date in long format. 12, January 2004

   Ensure that the date object holds a valid date. Otherwise display appropriate error message.

3. Write a program to illustrate Class Variables.

4. Use the special method **__del__** with the “**Person**” and the “**Date**” class and record your observation.

Compiled By : *Mohammed Mukthar Ahmed*

5. Write a program to illustrate Inheritance.

6. Create a class by name “Employee” which inherits the “Person” class attributes and methods, besides adds additional attributes like designation, and salary.
7. Write a program to illustrate multiple inheritance.

8. Using the name mangling mechanism, perform information hiding.

9. Perform the **isinstance( )** and **issubclass( )** tests on the Employee and Person objects.

Python Programming

Practice – 9

Overview

This practice reinforces us to understand and use some of the most common Standard Libraries available in Python.

- *Interface to Operating System.*
- *Perform different file tests.*
- *Perform string manipulation using Regular Expression.*
- *Perform Date and Time manipulation.*
- *Perform measurement. Etc.*

We shall put these elements in creating Python programs in this practice session.

Hands On

1. Create a Operating System Interface with the following options

```
Operating System Options
--------------------------------
1. Clear Screen
2. Date
3. Time
4. Current Directory
5. Quit
```

   Accept the user’s choice and perform appropriate operating system actions.

2. Accept absolute path from the user and find the following:

```
Path Manipulation
----------------------
1. Drive Name
2. Directory Name
3. File Name
4. Extension
5. Quit
```

3. Accept absolute path from the user and perform the following file test:

```
F I L E   T E S T
---------------------
1. Exists
2. Is Directory
3. Is File
4. Size If File
5. Quit
```

4. Use the file globing mechanism and display the size of all python regular file is the specified absolute path.

5. Use the sys module and find the following:

```
System  Information
------------------------
1. Version Number
2. Platform
3. Name of Executable
4. Quit
```

6. Use the string module and perform the following string operations for a given string:

```
String   Operations
----------------------
1. Find a Pattern
2. Count of Pattern
3. Find and Replace
4. Quit
```

7. Write a program to illustrate the use of “re” module.

8. Get the system date and display the date in the following formats:

   a) ANSI Format : yy/mm/dd
   b) British Format : dd-mm-yyyy
   c) American Format : mm-dd-yy
   d) Long Format: DD-MMM-YYYY
   e) Character Day of the Week

9. Accept the date of birth from the user and find the users age in years.