Unit 2

DATA, EXPRESSIONS, STATEMENTS

1. Name the four types of scalar objects Python has. **(AU Jan 2018, R2017)**

The commonly used scalar types in Python are:

Integers (type int), floating point numbers (type float), strings (type str), Booleans (type bool) and lists are a set of scalar objects in Python.

2. What is a Tuple? How literals of type tuple are written? Give example. **(AU Jan 2018, R2017)**

A tuple may be defined as a finite, static list of numbers or string. It contains immutable sequence of values separated by commas. The values can be of any type, and they are indexed by integers. Literals of type tuple are enclosed within parentheses ( ).

Example:

```
>>> t = ('a', 'b', 'c', 'd', 'e')
>>> max(5,8,9)
9
```

3. What is interpreter mode in Python?

The interpreter mode is the mode where the scripted and finished .py files are run in the Python interpreter.

4. What is interactive mode in Python?

The interactive mode is a command line shell which gives immediate feedback for each statement, fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole.

5. Define Keyword.

Keywords are reserved words that have predefined meanings in python. They cannot be used as ordinary identifiers and must be spelled exactly as they are written python3 has 33 keywords.

6. Name five primitive data types in Python.

Five primitive data types in Python are: Numbers, String, List, Tuple and Dictionary.

7. What does ‘immutable’ mean; which data type in python are immutable?

An object whose state or value cannot be changed in place is said to be immutable type. So, a sequence that is immutable sequence is one that cannot change. For example, Strings and Tuples are immutable.

8. What are escape sequences. Write any four escape sequence in Python.

Escape sequences are non printable characters. It consists of backslash followed by a character both are enclosed within single quotes.

\\	Backslash(\)
\’	Single quote(’)
\”	Double quote(”)
\a	ASCII Bell (BEL)

9. Define Value.

A value is the basic units of data, like a number or a string that a program manipulates.

10. What are the types used in Python.

Types is a category of values. Integers (type int), floating point numbers (type float), strings (type str), Booleans (type bool) and lists are a set of predefined data types in python. They are called as built-in-types.

11. What is boolean type in Python.

A Boolean type represents special values True and False. The most common way to produce a Boolean value is with a relational operator

12. How can you create a complex literal in Python?

Complex literals can be created by using the notation x + yj where x is the real component and y is the imaginary component. For example, >>> 1j * 1j which produces the result as: (–1 + 0j).

13. What is a variable?

A variable is an identifier, which holds a value. In programming, we assign a value to a variable. Technically, a variable is a reference to a computer memory, where the value is stored.

14. List out the rules to be followed in naming variable.

- Variable names can be at any length.
- They can contain both letters and numbers, but they can't begin with a number.
- Both uppercase letters and lower case letters can be used.
- The underscore (_) character can appear in a name. It is often used in names with multiple words.
- A variable name cannot be any one of the keywords.

15. What is expression.

An expression is a combination of values, variables, and operators. A value by itself is considered as an expression. Expressions, most commonly, consist of a combination of operators and operands.

Example: 5 + (8 * k)

16. Give example for simultaneous assignment statements.

Simultaneous Assignment Statement

Python permits any number of variables to appear on the left side separated by commas. The same number of expressions must then appear on the right side, again separated by commas.

Example:

```
 >>> x = 10
>>> y = 5
>>> sum, diff, prod = x + y, x – y, x
>>> sum
15
>>> diff
5
>>> pr
50
```

17. What is a tuple. Give example.

A tuple is a immutable sequence of values. They are comma separated list of values. The value can be any type, and they are indexed by integer.

Example:

```
>>> t = ‘a’, ‘b’, ‘c’, ‘d’, ‘e’
>>> t = (‘a’, ‘b’, ‘c’, ‘d’, ‘e’)
```

18. How to create a tuple.

Creating a tuple:

i) To create a tuple with a single element, final comma is to be included.

```
>>> t1 = ‘a’,
```

ii) Another way to create a tuple is the built-in function tuple. With no argument, it creates empty tuple:

```
>>> t = tuple()
>>> t
()
```

iii) If the argument is a sequence (string, list or tuple), the result is a tuple with the elements of the sequence:

```
>>> t = tuple(‘Jovita’)
>>> t
(‘J’, ‘o’, ‘v’, ‘i’, ‘t’, ‘a’)
```

19. How can we declare variables in Python?

Python variables do not have to be explicitly declared to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

20. How bracket operator used in tuple.

The bracket operator indexes an element.

```
>>> t = (‘a’,‘e’,‘i’,‘o’,‘u’)
>>> t[0]
‘a’
>>> t[3]
‘o’
```

The slice operator selects a range of elements.

```
>>> t[1:3]
(‘e’, ‘i’)
```

21. Give the precedence of operators in python.

- Parentheses have the highest precedence and can be used to force an expression to evaluate in the order.

  Example: 5 * (9 − 3) = 30 Since expressions in parentheses are evaluated first.

- Exponentiation has the next highest precedence.

  Example: 1 + 2 ** 3 = 9, not 27 and
  2 * 3 ** 2 = 18, not 36.

- Multiplication and Division have higher precedence than Addition and Subtraction.

  Example: 2 * 3 – 1 = 5, not 4 and
  6 + 4/2 = 8, not 5

22. What is a module?

A Python module is a file containing Python definitions and statements. The Python Standard Library is a collection of built-in modules, each providing specific functionality beyond what is included in the core part of python.

23. Define function.

A function is a named sequence of statements that performs a computation function may or may not take arguments and may or may not produce a result.

24. What are the advantages of using function?

Advantages of using functions in Python:

1. Reduction in code redundancy
2. Enabling code reuse
3. Better readability
4. Easy to debug and testing
5. Improved maintainability.

25. Give the syntax of function definition.

Syntax of function definition

```
def function_name(parameters):          // function header
Statement(s)                             // function body
```

26. Define parameter and argument.

- Parameter: A name used inside a function to refer to the value passed as an argument.

- Argument: A value provided to a function when the function is called. This value is assigned to the corresponding parameter in the function.

ANNA UNIVERSITY 16 MARK QUESTIONS WITH ANSWERS

1. What is a numeric literal? Give examples. (4)
(AU Jan 2018, R2017)

Refer Page No.: 92

2. Appraise the arithmetic operators in Python with an example (12)
(AU Jan 2018, R2017)

Refer Page No.: 100

3. Outline the operator precedence of arithmetic operators in Python (6)
(AU Jan 2018, R2017)

Refer Page No.: 107, 108

4. Write a Python Program to exchange the value of two variables (4)
(AU Jan 2018, R2017)

Refer Page No.: 120

5. Write a Python program using function to find the sum of first ‘n’ even numbers and print the result. (6)
(AU Jan 2018, R2017)

```
  def sum(n):
        i = 0
        sum = 0
        while i < n:
                  sum = sum + i
                  i = i + 2
return sum

n = int(input('Enter the limit '))
print(n)
result = sum(n)
print('Sum of even numbers = ',result)
```

6. Explain the following with suitable example.

   i) Tuple assignment.
   ii) Precedence of operators.

   Refer Page No.: 106, 107

7. Define function. Explain function definition and its uses with example.

   Refer Page No.: 111, 112

8. Write a Python program to test for leap year.

```
year = int(input("Enter a year: "))
if (year % 4) = = 0:
    if (year % 100) = = 0:
if (year % 400) = = 0:
    print("{0} is a leap year".format(year))
else:
    print("{0} is not a leap year".format(year))
else:
    print("{0} is a leap year".format(year))
else:
    print("{0} is not a leap year".format(year))
```