SOLVED QUESTIONS

1. Python has certain functions that you can readily use without having to write any special code. What types of functions are these?

Ans. The predefined functions that are always available for use are known as Python's built-in functions. Examples of some built-in functions are: len(), type(), int(), input(), etc.

2. What is a Python module? What is its significance?

Ans. A "module" is a chunk of Python code that exists in its own (.py) file and is intended to be used by Python code outside itself. Modules allow one to bundle together code in a form in which it can be easily used later. Modules can be "imported" in other programs so that the functions and other definitions in imported modules become available to code that imports them.

3. What are docstrings? How are they useful?

Ans. A docstring is just a regular Python triple-quoted string that is the first thing in a function body/a module/a class. When executing a function body (or a module/class), the docstring doesn't do anything like comment, but Python stores it as part of the function documentation. This documentation can later be displayed using the help() function.

So, even though docstrings appear like comments (no execution), they are different from comments.

4. What happens when Python encounters an import statement in a program? What would happen if there is one more import statement for the same module, already imported in the same program?

Ans. When Python encounters an import statement, it does the following:

(i) Code of the imported module is interpreted and executed.

(ii) Defined functions and variables created in the module are now available to the program that imported the module.

(iii) For imported module, a new namespace is set up with the same name as that of the module.

Any duplicate import statement for the same module in the same program is ignored by Python.

5. What is an argument? Give an example.

Ans. An argument is data passed to a function through function call statement. *For example*, in the statement print(math.sqrt(25)), the integer 25 is an argument.

6. Python has certain functions that you can readily use without having to write any special code. What type of functions are these?

Ans. The predefined functions that are always available for use are known as Python's built-in functions.

Examples of some built-in functions are: input(), type(), int(), eval(), etc.

7. What is the utility of built-in function help()?

Ans. Python's built-in function help() is very useful. When it is provided with a program name or a module name or a function name as an argument, it displays the documentation of the argument as help. It also displays the docstring within its passed-argument's definition.

For example,

```
>>>help(math)
```

will display the documentation related to module math.

It can even take function name as argument,

For example,

```
>>>help(math.sqrt())
```

The above code will list the documentation of math.sqrt() function only.

8. Write a function that:

(i) Asks the user to input a diameter of a sphere (centimetres, inches, etc.)

(ii) Sets a variable called radius to one-half of that number.

(iii) Calculates the volume of a sphere using this radius and the formula:
Volume = 4/3 pi*r*3

(iv) Prints a statement estimating that this is the volume of the sphere; include the appropriate unit information in litres or quarts. Note that there are 1000 cubic cm in a litre and 57.75 cubic inches in a quart.

(v) Returns this same amount as the output of the function.

Ans.

```
def compute_volume():
    import math
    diameter = float(input("Please enter the diameter in inches:"))
    radius = diameter/2
    Volume_in_cubic_inches = ((4/3) * math.pi) * (radius ** 3))
    Volume_in_quarts = Volume_in_cubic_inches/57.75
    print("The sphere contains", Volume_in_quarts, 'quarts')
    return(Volume_in_quarts)
```

9. What will be the output of the following code?

```
def interest (prnc, time=2, rate = 0.10):
    return (prnc * time * rate)
print(interest(6100, 1))
print(interest(5000, rate =0.05))
print(interest(5000, 3, 0.12))
print(interest(time = 4, prnc = 5000))
```

Ans.
```
610.0
500.0
1800.0
2000.0
```

10. What is the utility of Python standard library's math module and random module?

Ans. (i) Math module: The math module is used for math-related functions that work with all number types except for complex numbers.

(ii) Random module: The random module is used for different random number generator functions.

11. What is a module list? Give at least two reasons why we need modules.

Ans. A module is a file containing Python definitions and statements. We need modules because of their following salient features:

1. A module allows code reuse
2. It makes the "main" program shorter and more readable.
3. It enables clearer code organization

12. Observe the following code and answer the question based on it.

```
# the math_operation module
def add (a,b):
    return a + b
def substract (a,b):
    return a - b
```

Fill in the blanks for the following code:

1. `math_operation`
   `# get the name of the module.`
2. `print (_____)`
   `# output: math_operation`
   `# Add 1 and 2`
3. `print (_____(1, 2))`
   `# output 3`

Ans. 1. input 2. math_operation__name__

3. math_operation.add

13. Consider the code given in Q.7 and on the basis of it, complete the code given below:

```
# import the subtract function
# from the math_operation module
```

1. `_____`
   `# subtract 1 from 2`
2. `print(_____(2, 1))`
   `# output : 1`
   `# Import everything from math_operations`
3. `_____`
   `Print(subtract (2, 1))`
   `# output: 1`
   `Print(add (1, 1))`
   `# output : 2`

Ans. 1. from math_operation import subtract 2. subtract

3. from math_operation import *

14. Why is the use of import all statements not recommended?

Ans. While using import all statements, you can overwrite functions and this can be dangerous, especially if you don't maintain that module.

15. How will you share global variables across modules? [HOTS]

Ans. To do this for modules within a single program, we create a special module, then import the config module in all modules of our application. This lets the module be global to all the modules.

16. What is a Python module?

Ans. A **module** is a script in Python that defines import statements, functions, classes, and variables. It also holds executable Python code. ZIP files and DLL files can be modules too. The module holds its name as a string that is in a global variable.

17. What is namespace in Python?

Ans. In Python, every name has a place where it resides and can be looked for. This is known as namespace. It is like a box where a variable name is mapped to the object placed. Whenever the variable is searched, this box will be searched to retrieve the corresponding object.

18. What are the rules for local and global variables in Python? [HOTS]

Ans. In Python, variables that are only referenced inside a function are called implicitly global. If a variable is assigned a new value anywhere within the function body, it is assumed to be local. If a variable is ever assigned a new value inside the function, the variable is implicitly local and you need to explicitly declare it as 'global'.

19. Define 'module' and 'package'. [HOTS]

Ans. Each Python program file is a 'module' which imports other modules like 'objects' and 'attributes'. A Python program folder is a 'package' of 'modules'. A package can have 'modules' or 'sub-folders'.

20. Why is a banner saying "RESTART" always displayed in Python module/program execution?

Ans. When you execute a program from the IDLE Editor, the interpreter gives a banner saying "RESTART", meaning that all the things you defined in any shell session so far are wiped clean and the program you are running starts afresh.

21. What is the output of the following piece of code?

```
#mod1
def change(a):
    b=[x*2 for x in a]
    print(b)
#mod2
def change(a):
    b=[x*x for x in a]
    print(b)
from mod1 import change
from mod2 import change
#main
s=[1,2,3]
change(s)
```

Note: Both the modules mod1 and mod2 are placed in the same program.

(a) [2,4,6] (b) [1,4,9]
(c) [2,4,6], [1,4,9] (d) There is a name clash

Ans. (d)

Explanation: A name clash is when two different entities with the same identifier become part of the same scope. Since both the modules have the same function name, there is a name clash.

22. Which of the following is not a valid namespace?

(a) Global namespace (b) Public namespace
(c) Built-in namespace (d) Local namespace

Ans. (b)

Explanation: During a Python program execution, there are as many as three namespaces—built-in namespace, global namespace and local namespace.

23. Which of the following is false about "import modulename" form of import?
(a) The namespace of imported module becomes part of importing module
(b) This form of import prevents name clash
(c) The namespace of imported module becomes available to importing module
(d) The identifiers in module are accessed as modulename.identifier

Ans. (a)
Explanation: In the "import modulename" form of import, the namespace of imported module becomes available to, but not part of, the importing module.

24. What is the output of the following piece of code?
```
from math import factorial
print(math.factorial(5))
```
(a) 120
(b) Nothing is printed
(c) Error, method factorial doesn't exist in math module
(d) Error, the statement should be: print(factorial(5))

Ans. (d)
Explanation: In the "from-import" form of import, the imported identifiers (in this case factorial()) aren't specified along with the module name.

25. Which of the following is not an advantage of using modules?
(a) Provides a means of reusing program code
(b) Provides a means of dividing up tasks
(c) Provides a means of reducing the size of the program
(d) Provides a means of testing individual parts of the program

Ans. (c)
Explanation: The total size of the program remains the same regardless of whether modules are used or not. Modules simply divide the program.

26. Consider the set of height measurements (in feet) of students in a class:

Student	Height
Rishabh	5.9
Anna	5.5
Rinku	6.1
Tapish	6.0
Ajay	7.2

Write a Python program to calculate the average height of students in the class and median value using Statistics module.

Ans.
```
# To calculate the average height of the students in a class import statistics
heights = [5.9, 5.5, 6.1, 6.0, 7.2]                    # the height data
avg = statistics.mean(heights)
print("Average height in the class:",avg)
median_value = statistics.median(heights)
print("Median value for heights:",median_value)
```

Output:
```
Average height in the class: 6.14
Median value for heights: 6.0
```

27. Write a Python program to guess a number between 1 and 9.

Ans.
```
import random
target_num, guess_num = random.randint(1, 10), 0
while target_num != guess_num:
    guess_num = int(input("Guess a number between 1 and 10 \
                          until you get it right:"))

    print(target_num)
    target_num = random.randint(1, 10)
Print('Well guessed!')
```