204205-MDSC102-ASSIGNMENT1

1. What is a variable in Python?
Ans. A variable is a name that stores a value in it.

2. How do you create a variable?
Ans. We create a variable, by giving a name to it and giving it a value.

3. How do you check the value within a variable?
Ans. If variable name is 'name' then, print(name)

4. How do you create multiple variables in a single statement?
Ans. var1, var2, var3 = val1, val2, val3

5. How do you create multiple variables with the same value?
Ans. var1 = var2 = var3 = value

6. How do you change the value of a variable?
Ans. var = value
var = newvalue

7. How do you reassign a variable by modifying the previous value?
Ans. var = newvalue

8. What does the statement `counter += 4` do?
Ans. It increments the value of counter by adding 4 to it, whenever its called.

9. What are the rules for naming a variable?
Ans. Mustnot start with a digit
Variable name shouldn't be same as any of the reserved keywords.
They are case sensitive

10. Are variable names case-sensitive? Do `a_variable`, `A_Variable`, and `A_VARIABLE` represent the same variable or different ones?
Ans. Yes, variable names are case-sensitive. The above mentioned names are all different to each other.

11. What is Syntax? Why is it important?
Ans. Syntax is the representation of a language in which the code is written in the appropriate syntax for it to work. Maintaining syntax is important as it will lead to error in code if its not followed.

12. What happens if you execute a statement with invalid syntax?
Ans. While execution of code, syntax error occurs which doesn't provide the outcome.

13. How do you check the data type of a variable?
Ans. print(type(name))

14. What are the built-in data types in Python?
Ans. Integers, Complex, Float, Boolean, Strings, Arrays, Tuples, Lists, Sets, Dictionary

15. What is a primitive data type?
Ans. Primitive data types are the pre-defined data types.

16. What are the primitive data types available in Python?
Ans. Integers, Complex, Float, Boolean, Strings, Arrays, Tuples, Lists, Sets, Dictionary

17. What is a data structure or container data type?
Ans. A container is a class or data structure whose instances are collection of other objects.

18. What are the container types available in Python?
Ans. Lists, Sets, Tuples, Dictionary

19. What kind of data does the Integer data type represent?
Ans. Integer data type represents numerical value i.e. numbers

20. What are the numerical limits of the integer data type?
Ans. There are no numerical limits of the integer data type i.e. max

21. What kind of data does the float data type represent?
Ans. Float data type represents floating point i.e. decimals

22. How does Python decide if a given number is a float or an integer?
Ans. If the number given is 2, then it is integer
If the number given is 2.34, then it is float

23. How can you create a variable which stores a whole number, e.g., 4 but has the float data type?
Ans. float var = 2 or var=2.0

24. How do you create floats representing very large (e.g., 6.023 x 10^23) or very small numbers (0.000000123)?
Ans. By exponential notation 6.023e23 and 123e-9

25. What does the expression `23e-12` represent?
Ans. 23 * 10^-12 i.e. 0.000000000023

26. Can floats be used to store numbers with unlimited precision?
Ans. No

27. What are the differences between integers and floats?
Ans. Integers are real-valued numbers and Floats are decimal numbers

28. How do you convert an integer to a float?
Ans.
```
var = 2
float(var)
```

29. How do you convert a float to an integer?
Ans.
```
var = 2.0
int(var)
```

30. What is the result obtained when you convert 1.99 to an integer?
Ans. 2

31. What are the data types of the results of the division operators `/` and `//`?
Ans. 2/3 = 0.666666 Floating Point division
2//3 = 0 Floor division

32. What kind of data does the Boolean data type represent?
Ans. Boolean data type represent true or false i.e 1 or 0

33. Which types of Python operators return booleans as a result?
Ans. Logical Operators

34. What happens if you try to use a boolean in arithmetic operation?
Ans. It converts to integers

35. How can any value in Python be covered to a boolean?
Ans. typecasting i.e. bool(var)

36. What are truthy and falsy values?
Ans. Non boolean values used in boolean

37. What are the values in Python that evaluate to False?
Ans. 0

38. Give some examples of values that evaluate to True.
Ans. string values

39. What kind of data does the None data type represent?
Ans. no values present i.e. NoneType

40. What is the purpose of None?
Ans. To find out whether the value is None or not.

41. What kind of data does the String data type represent?
Ans. characters, numbers, symbols

42. What are the different ways of creating strings in Python?
Ans. varname = “value” or ‘value’, for example x = “sairam” & “““data””” for multi line strings.

43. What is the difference between strings creating using single quotes, i.e. `'` and `'` vs. those created using double quotes, i.e. `\"` and `\"`?
Ans. There is no difference between strings created using single quotes and double quotes

44. How do you create multi-line strings in Python?
Ans. using “““data”””

45. What is the newline character, `\\n`?
Ans. newline character is represented by \n and the data after \n will be present in next line.

46. What are escaped characters? How are they useful?
Ans. Characters with \n are known as escaped characters. They are useful for escape sequence to include both quotes in a string or to include more than one line in the string.

47. How do you check the length of a string?
Ans. x = ‘sairam’ then print(len(x))

48. How do you convert a string into a list of characters?
Ans. x = “sairam” then to conver this: y = list(x)

49. How do you access a specific character from a string?
Ans. By printing out in required index value i.e. print(str[1])

50. How do you access a range of characters from a string?
Ans. By printing the required range using index values i.e. print(str[2:5])

51. How do you check if a specific character occurs in a string?
Ans. x = “sairam” then print(‘s’ in x)

52. How do you check if a smaller string occurs within a bigger string?
Ans. If smaller string is x = ‘sairam’ and bigger string is z = sairamsrv; then print(x in z)

53. How do you join two or more strings?
Ans. x = sairam, y = srv then print(x+y)

54. What are \"methods\" in Python? How are they different from functions?
Ans. Method is a function which is a part of class. Methods are part of class whereas functions doesn’t belong to class unless its a function.

55. What do the `.lower`, `.upper` and `.capitalize` methods on strings do?
Ans. .lower convert the characters in strings to lowercase
.upper conver the characters in strings to uppercase
.capitalize convert the first letter in a string to uppercase

56. How do you replace a specific part of a string with something else?
Ans. x = ‘sairam’ then x.replace(‘sai’,‘you’)

57. How do you split the string \"Sun,Mon,Tue,Wed,Thu,Fri,Sat\" into a list of days?
Ans. days = "Sun,Mon,Tue,Wed,Thu,Fri,Sat" then days.split(‘,’)

58. How do you remove whitespace from the beginning and end of a string?
Ans. x=‘ sairam ’ then x.strip()

59. What is the string `.format` method used for? Can you give an example?
Ans. To add new characters in a string at required position, .format method is used.
x=‘my name is {}’ then x.format(‘srv’)

60. What are the benefits of using the `.format` method instead of string concatenation?
Ans. Using .format we can place it into particular position whereas string concatenation adds at the end of the string.

61. How do you convert a value of another type to a string
Ans. By typecasting, x = 40 then str(x)

62. How do you check if two strings have the same value?
Ans. x = ‘sairam’, y=‘hello’ then print(x == y)

63. Where can you find the list of all the methods supported by strings?
Ans. The list of all the methods supported by strings can be found in string class.

64. What is a list in Python?
Ans. A python list is an ordered, mutable array of objects.

65. How do you create a list?
Ans. my_list = [‘a’,‘5’]

66. Can a Python list contain values of different data types?
Ans. Yes

67. Can a list contain another list as an element within it?
Ans. my_list = [‘a’,‘5’], new_list = [my_list,7] then new_list will be printed as [['a', 5], 7]

68. Can you create a list without any values?
Ans. Yes

69. How do you check the length of a list in Python?
Ans. print(len(my_list))

70. How do you retrieve a value from a list?
Ans. By indexing i.e. print(my_list[1])

71. What is the smallest and largest index you can use to access elements from a list containing five elements?
Ans. smallest index is 0 and largest index is 4

72. What happens if you try to access an index equal to or larger than the size of a list?
Ans. IndexError: list index out of range occurs

73. What happens if you try to access a negative index within a list?
Ans. Elements will be accessed from backwards of a list starting from -1

74. How do you access a range of elements from a list?
Ans. print(my_list[2:4])

75. How many elements does the list returned by the expression `a_list[2:5]` contain?
Ans. 5

76. What do the ranges `a_list[:2]` and `a_list[2:]` represent?
Ans. a_list[:2] represents the first two elements and a_list[2:] represents all the elements starting from the third one.

77. How do you change the item stored at a specific index within a list?
Ans. my_list[2] = 6

78. How do you insert a new item at the beginning, middle, or end of a list?
Ans. At the beginning: my_list.insert(0,5)
At the middle: after knowing the lenght of the list, take middle value and follow the above.
At the end: my_list.append(11)

79. How do you remove an item from a list?
Ans. my_list.pop()

80. How do you remove the item at a given index from a list
Ans. my_list.pop(2)

81. How do you check if a list contains a value?
Ans. print(7 in my_list)

82. How do you combine two or most lists to create a larger list?
Ans. my_list.extend(new_list)

83. How do you create a copy of a list?
Ans. new_list = my_list.copy()

84. Does the expression `a_new_list = a_list` create a copy of the list `a_list`?
Ans. No

85. Where can you find the list of all the methods supported by lists?
Ans. The list of all the methods supported by strings can be found in list class.

86. What is a Tuple in Python?
Ans. A tuple is an immutable list i.e. cannot be appended, extended or remove elements.

87. How is a tuple different from a list?
Ans. Tuple is immutable and list is mutable array of objects.

88. Can you add or remove elements in a tuple?
Ans. No

89. How do you create a tuple with just one element?
Ans. t = (‘hi,’)

90. How do you convert a tuple to a list and vice versa?
Ans. Tuple to a list: t = (1,5,4) then l = list(t)
List to a tuple: l = [1,5,4] then t = tuple(l)

91. What are the `count` and `index` method of a Tuple used for?
Ans. count – number of times an element present in tuple
index – to access elements of a particular index

92. What is a dictionary in Python?
Ans. The dictionary is a collection of key-value pairs which are mutable objects.

93. How do you create a dictionary?
Ans. newdict = {'Age': 20, 'Height':180}

94. What are keys and values?
Ans. Keys are the identifiers and values are the data stored in the key

95. How do you access the value associated with a specific key in a dictionary?
Ans. newdict['Age']

96. What happens if you try to access the value for a key that doesn't exist in a dictionary?
Ans. KeyError is thrown

97. What is the `.get` method of a dictionary used for?
Ans. used to retrive the value, given a key if it exists

98. How do you change the value associated with a key in a dictionary?
Ans. newdict['Age'] = 24

99. How do you add or remove a key-value pair in a dictionary?
Ans. add key-value pair: newdict[‘Sex’] = ‘male’
remove key-value pair: del newdict['Sex']

100. How do you access the keys, values, and key-value pairs within a dictionary?
Ans. To access keys: newdict.keys()
To access values: newdict.values()
To access key-value pairs: newdict.items()