ASSIGNMENT - I

① Write about input/output statements in python with suitable examples.

Input:

* A developer might want to take user input at some point in the program.
* To take input from user there exist a built-in function in python.

Syntax: input (' prompt ')

Where prompt is an optional string that is displayed on the screen at the time of taking input.

Example:

```
name = input("Enter your name: ")
print (name)
print (type(name))
```

O/p: Enter your name : Saketh
Saketh
<class 'str'>

* By default any input given in python is taken as string.
* To convert it to any other data type we have to convert it to any other data type explicitly.
* For example to convert input into int we have to use int() method.

Example: ① a = input ('enter a number:')

```
x = int (a)
print (x)
```

O/p: enter a number: 5
5

② x = int (input ("Enter a number:")

```
print (type (x))
print (x)
```

O/p: <class 'int'>
9

* A programmer uses split() function while giving input inorder to separate the input.

Example:

```
a,b = [int(input('Enter 1st number')), int(input(
    'Enter 2nd number')).split(',')]
print ('The product :' a*b)
```

O/p: Enl

Example:

```
a,b = [int(x) for x in input("Enter 2 numbers:").
      split(',')]
print ('The product:' a*b)
```

O/p: Enter 2 numbers: 10,20
The product : 200

Output:

* Python provides the print() function to display output to the standard output devices

Syntax: print ("Statement", variable)

Ex: ① print ("Hello")

O/p: Hello

② a=5
print ("value of a:" a)
O/p: value of a: 5

* We can also print variables using format strings. We use % operator. The formatting using % is similar to that of 'printf' in the C programming language.

- %d - integer
- %f - float
- %s - string
- %x - hexadecimal
- %o - Octal
- %i - integer

Ex:

```
num = int (input("Enter a value:"))
add = num+5
print (" The sum is %d "%add)
```

O/p: Enter a value: 50
The sum is 55

② Explain the following with an example.

a. comparision Operators

b. Assignment Operators

c. Logical Operators

* Operators are used to perform operation between operands.
* There are various operators in python.

a. Comparision Operators:

* It is also known as Relational Operators.
* These operators are used to compare operands.
* '<', '>', '<=', '>=', '==', '!=' are the relational operators.
* Comparision Operators compares numbers, strings, boolean values and also numbers with strings.

Examples:
```
a,b = [int(input('Enter 2 numbers').split()]
print (a>b)
print (a<b)
print (a<=b)
print (a>=b)
print (a==b)
print (a!=b)
```
o/p: Enter 2 number 5 4

True
False
False
True
False
True

* It always returns output in boolean type.

Example:
```
x = 'Python'
y = 'Computer'
print (x<y)
print (ord(y)>print(p))
print (10>True)
print (True<False)
print (10=='CSE')
```
o/p: True
False True
True
False
False

* In comparing strings it considers the first letter and if first letter is same it goes on comparing all letters one by one
* We cannot use '>', '<' symbols inbetween numbers and strings comparision

Example: print (10>='CSE')
o/p: Error

* We can also chain operators

Ex: print (5<9>7)

O/p: True

* In such case every case must be true to get output true.

b. Assignment Operator:

* Assignment operators are used to assign a value to a variable and to increase a variables assigned value.

* '=', '+=', '-=', '*=', '/=', '//=', '**=' are the assignment operators

Example:

```
a = 20
b = 20
print (a+=b)
print (a-=b)
print (a*=b)
print (a/=b)
print (a//=b)
print (a**=b)
```

O/p: 30
20
200
20.0
2.0
100.0

Operator	Description	Syntax (examples)
=	Assign value for right side of expression to left side operand	x = y+z
+=	Add and Assign: Add right side operand with left side operand and assign to left operand.	a+=b
-=	Subtract and Assign: Subtract right operand from left operand and then assign to left operand: True if both operands are equal.	a-=b
*=	Multiply and assign: Multiply right operand with left operand and then assign to left operand	a*=b
/=	Divide and assign: Divide left operand with right and then assign to left operand	a/=b
//=	Divide and assign: Divide left operand with right and assign floor value	a//=b

	to left Operand	
** =	Exponent and: Calculate exponent value using operands and assign value to left operand	a**=b

C. Logical Operators:

* In python we use logical and, logical or, logical not operators as logical operators
* There are no special symbols for these operators. We use and, or, not for using these in program.
* We use logical operators to compare 2 expressions.

Operator	Description	Example
and	When both the expressions are true it returns true	print(5<6 and 6<7) o/p: True
or	When either of the expressions are true it returns true else it returns false	print(8>9 and 8<9) o/p: True
not	It prints negation of value of expression	print(not(8>9)) o/p: True

③ What are the 4-build-in numeric data types in Python? Example

* The four built-in numeric datatypes are
  - (i) int
  - (ii) Float
  - (iii) Complex
  - (iv) Boolean

int: Integers i.e., positive, negative whole numbers ~~complex~~ comes under this type.

* Fractional part is not included in this data type.
* Integers can be binary, Octal, and hexadecimal values
* In python there is no limit to how long an integer value can be.

Ex:

```
a = 5
print("Type of a: ", type(a))
```

o/p: Type of a: <class 'int'>

Float: This value is represented by float class. It is a real number with floating point representation.

* It is specified by a decimal point.
* Float can also be represented by numbers in scientific notation which contains exponents.
* Both a lower case e or an upper case E can be used to define floats in scientific notation.

Ex:
```
a = 2.8
print ("Type of a:", type(a))
```
O/p: Type of a: <class 'float'>

Complex:

* Complex number is represented by complex class.
* It is specified as (real part) + (imaginary part)j. where $j^2 = -1$
* To print real part of complex number we use
  print (variable.real)
* To print imaginary part of complex number we use
  print (variable.~~com~~imag)
* Any type of number can be used in real part but only decimal number can be used in ~~real~~ imagin -ary part.

Ex:
```
a = 2 + 3j
print (a.real)
print (a.imag)
print ("Type of a:", type(a))
```
O/p:
```
2
3
Type of a: <class 'complex'>
```

Bool:

* It is used to represent boolean type.
* It defines only 'True' and 'False'.
* True is considered as '1' and False is considered as '0'.

Ex:
```
a = 6
b = 3
c = a<b
print (c)
print ("Type of c:", type(c))
```
O/p: False
Type of c: <class 'bool'>

---

④ Mention differences between set and frozen set.

Set: * Set is a datatype in python that allows storing lots of mutable data into a single variable

* The elements of the sets are unordered, there is no index number associated with them.
* Set are mutable just like a list which means, once a set is defined we can modify and update it later.
* No duplicate values are allowed in set.
* add() is used to add elements in set
* remove() is used to delete elements in a set.

Frozen Set:

* Frozen means unmoving or fixed.
* The frozenset() is an built-in function in python that takes an iterable object as input and makes immutable
* Frozenset is a new class that has the characteristics of a set, but its elements cannot be changed once assigned.

Frozenset Vs Set:

* Set is a most basic level datatype, It supports all the method operations of the set such as add(), remove(), and so on.

```
Ex: A={1,2,3,4}
    A.add(8)
    print(A)
    O/p: {1,2,3,4,8}
```

* Frozen set is immutable, it does not support any operations like add(), remove(), and so on

```
Ex: A = frozenset([1,2,3,4])
    A.add(8)
    print(A)
O/p: File "<string>", line2, in <module>
     AttributeError: 'frozenset'
     object has no attribute 'add'
```

⑤ Explain in detail about python type conversion and type casting with example.

Type Conversion:

* Conversion of a data type into another data type is called as type conversion.
* There are 2 types of type conversions
  1. Implicit Type Conversion.
  2. Explicit Type Conversion.
* Implicit type conversion means that the python automatically converts it data type whenever required

```
Ex: a=12
    b=3.5
    c=a+b
    print(c)
    print(type(a))
    print(type(b))
    print(type(c))
O/p: 15.5
<class 'int'>
<class 'float'>
<class 'float'>
```

* Explicit conversion means manually user have to convert data type of a variable as it doesnot

support. for example in addition of string and integer we need to convert into int for string ~~of the~~ Otherwise it displays error.

Ex:

```
a = "12"
b = 3
c = int(a)
d = b+c
print (d)
print (type(a))
print (type(b))
print (type(c))
print (type(d))
```

O/p: 15

~~string~~

```
<class 'str'>
<class 'int'>
<class 'int'>
<class 'int'>
```

* If a data type of a variable is converted by compiler it is known as type conversion.
* If a data type of a variable is converted by user manually with functions like int(), float(), str() then it is known as type casting.