Chapter 7

A.

1. D
2. D
3. C
4. A
5. D
6. A
7. A
8. A
9. A
10. C
11. D

A.

1. False
2. False
3. False
4. False
5. False
6. True
7. True
8. True
9. True
10. True
11. True
12. True
13. False
14. True
15. False

B.

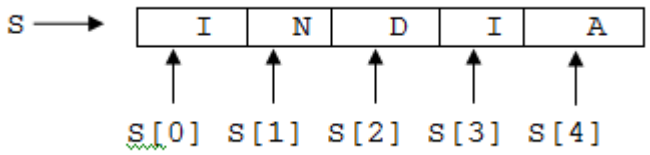
1. In python Strings are sequences of characters. Strings are immutable. This means once defined they cannot be changed. Strings in python can be written in double quotes, single quotes or triple quotes.

2. String can be created by making use of constructor as follows

```
S1=str()                #Creates an Empty string Object
```

```
S2=str("Python")        #Creates a String Object for Hello
```

3. Index operator is used to access the characters from given string using [] operator. The first character of string is stored at 0^{th} position. Whereas, the last character of a string is stored at position one less than of length of the string. The graphical representation of how the string is stored is shown as follows.



Example:

```
>>> str = 'Python'
>>> str[3]
'h'
>>> str[0]
'P'
>>> str[5]
'n'
```

4.

a. 29
b. r
c. m
d. True
e. 'me to'
f. 3
g. T
h. 'Welcome to Python Programming Welcome to Python Programming'

5. The following program illustrates traversing every third character of a string.

```
str = 'INDIA WON GOLDM MEDAL IN SWIMMING'
print(str[:3])
```

6. Yes we can traverse all the elements of a string using while loop. The following program illustrates the use of while loop to traverse all characters within a string.

```
S = 'INDIA WON GOLD MEDAL IN SWIMMING'
index=0
while index<len(S):
    print(S[index],end="")
    index=index+1
```

7. Strings are said to be immutable. Immutable means once the string is created we cannot change the contents of the string.

8. The slicing operator returns a subset of a string called **slice** by specifying two indices, viz. **start** and **end**. The syntax used to return a subset of a string is:

String_Variable_Name[Start_Index: End_Index]

Examples:

```
>>> a='Hello World'
>>> a[4:8]
'o Wo'
>>> a[6:10]
'Worl'
>>> a[6:11]
'World'
```

Thus, slicing operator returns subset of a string.

9. Subset of a string can be obtained by making use of slicing operator as discussed above in example 8.

10. Step size can be used for example programmer wants to select every second character from a string. In such cases programmer can make use of step size. The syntax of using step sizing in string slicing is as follows

Syntax:

Name_of_Variable_of_a_String[Start_Index:End_Index:Step_Size]

Example:

```
>>>S="IIT-BOMBAY"
>>> S[0:len(S):2]
>>>'ITBMA'
```

11. Operators such as **==,<,>,<=,>=**and **!=** are used to compare the

>>> A="Python"
>>> B="Qython"
A < B
True
	Returns true since ascii value of Q is more than the ascii value of P.

>>> A= "Python"
>>> B= "Python"
>>> A == B
True
	Check if two charters are equal using equality operator.

>>> A="Python"
>>> B="python"
>>> A!=B
True
	Not equal returns ture if both the strings are not same.

h
on compares strings by comparing their corresponding characters.

12. The **format ()** method is used to display the output in most best representable form.

13. The **split()** method is used to break the string. It returns a list of all the words in the string.

Example:

```
>>>Str1="C C++ JAVA Python”
>>>Str1.split()
['C,C++,JAVA,Python']
```

14.

Methods of Str Class for Testing its Character	Meaning
bool isalnum()	Returns true if characters in this string are alphanumeric and there is at least one character.
Example: >>>S="Python Programming" >>>S.isalnum() False	
bool isalpha()	Returns true, if the characters in this string are alphabetic, and there is at least one character.

Example:
>>> S="Programming"
>>>S.isalpha()
True
>>> S="1Programming"
>>>S.isalpha()
False	

bool isdigit()	Returns true, if the characters in this string contains, only digits.

Example:
>>> Str1="1234"
>>> Str1.isdigit()
True	

bool islower()	Returns true if all the characters in a string are in lowercase.
Example: >>> S="hello" >>>S.islower() True	
bool isupper()	Returns true if all the characters of this string are in uppercase.
Example: >>> S="HELLO" >>>S.isupper () True	

bool isspace()	Returns true, if this string contains only white space characters.

Example:

```
>>> S=" "
>>>S.isspace()
True
>>> Str1="Hello Welcome to Programming World"
>>> Str1.isspace ()
False
```

15.

Formatting a character meaning placing string or characters to one of the said alignment i.e. left, right, center or justify. Following in build functions are supported by python for performing formatting.

str center(int width)	Returns a copy of this string centered in a field of the given width.
str ljust(int width)	Returns a string left justified in a field of the given width.
str rjust(int width)	Returns a string right justified in a field of the given width.

Programming Assignments

1.

```
def Count_Upp_Lower(str):
    count = 0
    count1 = 0
    for x in str:
        if x.isupper():
            count = count + 1
        else:
            count1 = count1 + 1
    print('Total Uppercase characters  are: ',count)
    print('Total Lowercase Characters are : ',count1)
str = input('Please Enter the String:')
Count_Upp_Lower(str)
Output
Please Enter the String:PythOn
Total Uppercase characters  are:  2
Total Lowercase Characters are :  4
```

2.

```
def Echo_Word(word):
    for i in word:
        print(word)
Echo_Word('abcd')
```

Output:

```
abcd
abcd
abcd
abcd
```

3.

```
def reversedWord(word):
    return word[::-1]

#Sample run
x = reversedWord('Python')
print(x)
```

Output:

```
nohtyP
```

4.

```
def startEndVowels(word):
    print('word = ',word)
    vowels = ('a','e','i','o','u','A','E','I','O','U')
    print('The word starts with a vowel?',word.startswith(vowels))
    print('The word ends with a vowel?',word.endswith(vowels))
#Sample run
startEndVowels('I Love U')
```

Output:

```
word =  I Love U
The word starts with a vowel? True
The word ends with a vowel? True
```

5.

```
def getVowels(word):
    print('word = ',word)
    print(' Vowels from the   string ',word)
```

```
    word = word.lower()
    lst = list()
    for i in word:
        if (i == 'a' or i == 'e' or i == 'i' or i == 'o' or i == 'u' ):
            lst.append(i)
    return lst
#Sample run
x = getVowels('INDIA')
print(x)
```

Output:

```
word =  INDIA
Vowels from the   string  INDIA
['i', 'i', 'a']
```

6.

```
def Bin_to_Dec(n):
   l =len(n)
   p = 0
   s = 0
   n = n[::-1]
   for x in n:
      z= (int(x)*(2**(p)))
      s = s + z
      p = p + 1
   print(s)
bin = (input('Enter the  binary Number:'))
Bin_to_Dec(bin)
```

Output:

```
Enter the  binary Number:1000
8
```