Introduction to Strings

- At the end of this class, students will be able to-
  - Use the variable type – String
  - Create and use Strings using String built – in functions

Data structure: str : string

- A string is a sequence of characters.
- Python directly supports a str type; but there is no character type.
- A string has zero or more characters
- Each character of a string can be referred to by a index or a subscript
- An index is an integer

- Access to an element based on index or position takes the same time no matter where the element is in the string – random access
- strings are immutable. Once created, we cannot change the number of elements – no append, no insert, no remove, no delete.
- Elements of the string cannot be assigned.

- A string can not grow and cannot shrink. Size of the string can be found using the function len.
- String is a sequence
- String is also iterable - is eager and not lazy.
- Strings cannot be nested.
- Strings can be sliced. This creates a new string.

PES UNIVERSITY

4 types of string literals or constants.

- a) single quoted strings
- b) double quoted strings
- c) triple quoted strings
- d) raw strings

single quoted strings or Double

- There is no diference between the two. In both these strings, escape sequences like \t, \n are expanded.
- These strings can span just a line – cannot span multiple lines.
- >>> print 'It\'s raining‘
- It's raining
- >>> 'It\'s raining' # Same string specified differently
- "It's raining"
- >>> print "\"hello\""
- "hello"
- >>> print '"\\" is the backslash' # "\" instead of "\\"
- "\" is the backslash

triple quoted strings

- create a string spanning multiple lines by using either three single quotes or three double quotes as delimiters.

```
a="hello\
world"
print(a)
b="""hello
world"""
print(b)
>>>
helloworld
hello
world
```

raw strings

- A rawstring is a string literal (prefixed with an r) in which the normal escaping rules have been suspended so that everything is a literal.
- >>> a=r"Hi\nHello"
- >>> print(a)
- Hi\nHello

Working with String

- a=“” #empty string
- create a new string by concatenation
- >>> a=a+"hello"

  >>> a

  'hello‘
- x=“Hello”
- x=x.upper() # new string
- HELLO

PES UNIVERSITY

Program using upper() and title()

```
name="mohandas karamchand gandhi"
#op  M K Gandhi
lst=name.split()
#lst=['mohandas','karamchand','gandhi']
_1name=""
for i in  lst[:len(lst)-1]:
            _1name=_1name+i[0]+" "
_1name=_1name+lst[-1]
print(_1name)
_1name=_1name.title()
print(_1name)
_1name=_1name.upper()
print(_1name)
```

Program using endswith

```
name=[
          "Bangalore Karnataka"
          "Mysore Karnataka"
          "Lucknow UP"
          "Kanpur UP"
          "Belgavi Karnataka"
          ]
for i in name:
          if i.endswith("Karnataka"):
                    print(i)
```

Program with index()

- s="a nation of the people by the people and for the people"
- >>> s.index("people")# Index of first p
- 16
- s.index("people",(s.index("people")+1))
- 30
- s.index("people",(s.index("people",(s.index("people")+1))+1))
- 49

Program with replace()

- pharse="""I felt happy because I saw the others were happy and because I knew I should feel happy, but I wasn't really happy"""
- print(pharse.replace("happy","sad"))
- print(pharse.replace("happy","sad",3))

Program with replace()

- s="hello world hello python hello program"
- >>> i=s.index("hello",s.index("hello")+1)
- >>> i
- 12
- >>> s[:i]+s[i:].replace("hello","bad")
- 'hello world bad python bad program'

Example:

```
s="This is a python program"
for i in s:
            print(s[5:])
print()
Op:::::?
```

PES UNIVERSITY

Example:

```
s="This is a python program"
for i in s.split():
            print(s[5:])
print()
Op:::::?
```

Example:

```
s="This is a python program"
for i in s.split():
        print(i[1:],end="")
        print(i[0],end=" ")
print()
 op::::?
```

```
s="This is a python program"
for ch in s:
        print(ch, end = "")
        print('p', end = "")
print()
Op:::::?
```

- Write a Python program that prompts the user for a list of integers, stores in another list only those values that are in tuple valid_values, and displays the resulting list.

```
valid_values = (1, 4, 8, 11, 16, 21)
num_list = []
empty_str = ''
print('Enter a series of integers,one per line (hit return when
   done)')
entry = input('Enter: ')
while entry != empty_str:
        num = int(entry)
        if num in valid_values:
                num_list.append(num)
        entry = input('Enter: ')

print(num_list)
```

- Write a program that is containing list of words and sort them from longest to shortest:

```
txt = 'Climb mountains not so the world can see you,
  but so you can see the world.'
words = txt.split()
words.sort(reverse=True,key=len)
print(words)
```

- Write a Python program to get a string made of the first 2 and the last 2 chars from a given a string.

```
n=input(“enter string”)

a=n[:2]+n[-2:]

print(a)
```

- Write a Python program to get a string from a given string where all occurrences of its first char have been changed to '$', except the first char itself.

```
n= input("enter string")
a=n[0]
print(a+n[1:].replace(a,"$"))
```

- Write a Python program to get a single string from two given strings, separated by a space and swap the first two characters of each string.

```
n= input("enter string")
# abc xyz Op::xycabz
a=n.split()
n1=a[0]
n2=a[1]
x=n2[:2]+n1[2:]
y=n1[:2]+n2[2:]
print(x+y)
```

- Write a Python program to remove the characters which have odd index values of a given string.

```
n= input("enter string")
r=''
for i in range(0,len(n)+1):
        if i%2==0:
                r=r+n[i]
print(r)
```

Summary

- A string is a sequence of characters.
- Python directly supports a str type; but there is no character type
- Strings are immutable