Python for Selenium

Conditional Statements

- If else
- elif

Iterative Statements

- for loop
- while loop

range()

```
print(list(range(10))) #[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(list(range(5,10))) #[5, 6, 7, 8, 9]

print(list(range(1,10,2))) #[1, 3, 5, 7, 9] Odd numbers
print(list(range(0,10,2))) #[0, 2, 4, 6, 8] Even numbers

print(list(range(10,1,-1))) #[10, 9, 8, 7, 6, 5, 4, 3, 2] Decrement

print(list(range(-10,-5))) #[-10, -9, -8, -7, -6]   Negatives
print(list(range(-10,-5,2))) #[-10, -8, -6] Negatives increment by 2
```

for loop

```
#Print 1..9 numbers
for i in range(10):
    print(i)

#Print Evens between 2..9 numbers
for i in range(2,10,2):
    print(i)

#Print Odds between 1..9 numbers
for i in range(1,10,2):
    print(i)

#Print numbers between 10..2 in descending order
for i in range(10,1,-1):
    print(i)
```

While loop

```
#Prints 1..9 numbers
i=1
while i<=10:
    print(i)
    i=i+1

#Prints 10..1 numbers in decending order
i=10
while i>=1:
    print(i)
    i=i-1
```

Jumping statements

- break
- continue

break & continue

```
#break
for i in range(1,10):
    if i==5:
        break
    print(i)
print("program exited")


#continue
for i in range(1,10):
    if i==5:
        continue
    print(i)
print("program exited")
```

Working with Numbers

- Number Types
- Number Type conversions
- Built-in functions on Number Type

Number Types

```
a=10     #Integer
b=20.5   #Float or Double
print(a)
print(b)
```

Number Type Conversion

- Type **int(x)** to convert x to a plain integer.
- Type **float(x)** to convert x to a floating-point number.

```
x=10
print(int(x))
print(float(x))

print(type(x))
print(type(int(x)))
print(type(float(x)))
```

max() & min() functions on Number type

- **max() -** Returns the largest of its arguments.
- **min()** - Returns the smallest of its arguments.

```
print("max of 80, 100, 1000:", max(80, 100, 1000))
print("max of -10, 10, 5:", max(-10, 10, 5))

print("min of 80, 100, 1000:", min(80, 100, 1000))
print("min of -10, 10, 5:", min(-10, 10, 5))
```

Python Strings

- Strings in python are contiguous series of characters delimited by single or double quotes. Python don't have any separate data type for characters so they are represented as a single character string.

```
#Creating strings
name = "John" # a string
mychar = 'S' # a character
print(name)
print(mychar)

#you can also use the following syntax to create strings.
name1 = str() # this will create empty string object
name2 = str("Scott") # string object containing 'newstring'
print(name1)
print(name2)
```

Strings in python are immutable

- Once string is created it can't be modified (immutable)
- id() : Every object in python is stored somewhere in memory. We can use id() to get that memory address.

```
str1="welcome"
str2="welcome"

print(id(str1),id(str2))   #57660416 57660416

str2=str2+"to python"
print(id(str1),id(str2))   #57660416 59955200
```

+ and * Operations on string

- String index starts from 0.
- + operator is used to concatenate string and * operator is a repetition operator for string.

```
str="welcome"
print(str+" to Python programming") # welcome to Python programming
print(str *3) #welcomewelcomewelcome
```

Slicing string

- We can take subset of string from original string by using [] operator also known as slicing operator.
- **Syntax**: s[start:end]
- this will return part of the string starting from index **start** to index **end-1** .

```
str="welcome"

print(str[1:3])  # el
print(str[2:4]) #  lc

#same
print(str[:6]) #welcom
print(str[0:6])  #welcom

print(str[2:]) #lcome
print(str[2:7]) #lcome

print(str[1:-1]) #elcome
print(str[1:-2]) # elco
print(str[2:-3]) #lc
```

ord() and chr() Functions

- **ord()** – function returns the ASCII code of the character.
- **chr()** – function returns character represented by a ASCII number.

```
print(ord('A')) #65
print(chr(65)) #A
```

len(),max() and min() Functions on Strings

```
print(len("hello")) #5
print(max("abc")) #c
print(min("abc")) #a
```

in and *not in* operators

- You can use in and not in operators to check existence of string in another string. They are also known as membership operator.

```
s1 = "Welcome"
print("come" in s1)# True
print("come" not in s1) #False
```

String comparison

- You can use (> , < , <= , <= , == , !=) to compare two strings.
- Python compares string lexicographically i.e using ASCII value of the characters.

```
print("tim" == "tie") #False
print("free" != "freedom") #True
print ("arrow" > "aron") #True
print ("right" >= "left") #True
print ("teeth" < "tee") #False
print ("yellow" <= "fellow") #False
print ("abc" > "") #True
```

Iterating string using for loop

- String is a sequence type and also iterable using for loop

```
s = "hello"
for i in s:
    print(i)
    print(s, end="\n")   # this is default behavior
    print(s, end="")   # print string without a newline
    print(s, end="foo")   # now print() will print foo after every string
```

Testing strings

String class in python has various inbuilt methods which allows to check for different types of strings.

Method name	Method Description
isalnum()	Returns True if string is alphanumeric
isalpha()	Returns True if string contains only alphabets
isdigit()	Returns True if string contains only digits
isidentifier()	Return True is string is valid identifier
islower()	Returns True if string is in lowercase
isupper()	Returns True if string is in uppercase
isspace()	Returns True if string contains only whitespace

```
s = "welcome to python"
print(s.isalnum()) #False
print("Welcome".isalpha()) #True
print("2012".isdigit()) #True
print("first Number".isidentifier())#False
print(s.islower()) #True
print("WELCOME".isupper()) #True
print(" ".isspace()) #True
```

Searching for Substrings

Method Name	Methods Description:
endswith(s1: str): bool	Returns True if strings ends with substring s1
startswith(s1: str): bool	Returns True if strings starts with substring s1
count(substring): int	Returns number of occurrences of substring the string
find(s1): int	Returns lowest index from where s1 starts in the string, if string not found returns -1
rfind(s1): int	Returns highest index from where s1 starts in the string, if string not found returns -1

```
s = "welcome to python"
print(s.endswith("thon")) #True
print(s.startswith("good")) #False
print(s.find("come")) #3
print(s.find("become")) #-1
print(s.rfind("o")) #15
print(s.count("o")) #3
```

Converting Strings

Method name	Method Description
capitalize(): str	Returns a copy of this string with only the first character capitalized.
lower(): str	Return string by converting every character to lowercase
upper(): str	Return string by converting every character to uppercase
title(): str	This function return string by capitalizing first letter of every word in the string
swapcase(): str	Return a string in which the lowercase letter is converted to uppercase and uppercase to lowercase
replace(old, new): str	This function returns new string by replacing the occurrence of old string with new string

```
s = "String in PYTHON"
s1 = s.capitalize()
print(s1) #String in python

s2 = s.title()
print(s2)#String In Python

s3 = s.lower()
print(s3) #string in python

s4 = s.upper()
print(s4) #STRING IN PYTHON

s5 = s.swapcase()
print(s5) #sTRING IN python

s6 = s.replace("in", "on")
print(s6) #String on PYTHON

print(s) #String in PYTHON
```