Chapter - 5: Sequence Data Types

Strings

- Strings in Python are arrays of bytes representing unicode characters.
- Python does not have a character data type, a single character is simply a string with a length of 1.
- Strings are enclosed characters in quotes. Python treats single quotes the same as double quotes.
- Strings are immutable means that the contents of the string cannot be changed after it is created

Creating String

```
var1 = 'Hello World!'
var2 = "Python Programming"
```

Access Items

- Square brackets can be used to access elements of the string.
- To access substrings, use the square brackets for slicing along with the index or indices to obtain your substring.

```
var1 = 'Hello World!'
var2 = "Python Programming"

print ("var1[0]: ", var1[0])
print ("var2[1:5]: ", var2[1:5])
```

Output –

```
var1[0]:  H
var2[1:5]:  ytho
```

Negative Indexing

Negative indexing means beginning from the end, -1 refers to the last item, -2 refers to the second last item etc.

String A	S	A	V	E		E	A	R	T	H
Positive Index	0	1	2	3	4	5	6	7	8	9
Negative Index	-10	-9	-9	-7	-6	-5	-4	-3	-2	-1

```
var2 = "Python Programming"
print ("var2 last Character: ", var2[-1])
```

Output : g

Range of Indexes (Slicing)

We can specify a range of indexes by specifying where to start and where to end the range.

```
var2 = "Python Programming"
print ("var2[2:5]: ", var2[2:6])
print ("var2[2:5]: ", var2[2:6:2])
```

Output : thon
Output : to

Range of Negative Indexes

Specify negative indexes if you want to start the search from the end of the string.

```
var2 = "Python Programming"
print ("var: ", var2[-4:-1])
print ("var: ", var2[-4:-1:2])
```

Output : min
Output : mn

Updating Strings

We can "update" an existing string by (re)assigning a variable to another string.

```
var2 = "Python Programming"
var2="Hello "+var2[:6]
print(var2)
```

Output : Hello Python

Loop Through a String

We can loop through the **String** items by using a for loop:

```
var2 = "Python Programming"
for x in list:
    print(x)
```

Check if Item String

To check if a certain phrase or character is present in a string, we can use the keywords in or not in.

```
txt = "The rain in Spain stays mainly in the plain"
x = "ain" in txt
print(x)
```

Output True

```
txt = "The rain in Spain stays mainly in the plain"
x = "ain" not in txt
print(x)
```

Output False

Length of Set

To determine how many items a **String** has, use the **len( )** function.

```
var="Python Programming"
print(len(var))
```

Escape Character

- To insert characters that are illegal in a string, use an escape character.
- An escape character is a backslash \ followed by the character you want to insert.

The escape character allows you to use double quotes when you normally would not be allowed:

```
txt = "We are the so-called \"Vikings\" from the north."
print(txt)
```

List of Escape Character

Code	Result
\'	Single Quote
\\	Backslash
\a	Bell or alert
\n	New Line
\r	Carriage Return
\t	Tab
\b	Backspace
\f	Form Feed
\ooo	Octal value
\xhh	Hex value

String Concatenation

To concatenate, or combine, two strings you can use the + operator.

```
a = "Hello"
b = "World"
c = a + b
print(c)                    # Hello World
```

String Methods

The **strip**( ) method removes any whitespace from the beginning or the end

```
a = "  Hello, World!  "
print(a.strip())          # returns "Hello, World!"
```

The **lower**( ) method returns the string in lower case:

```
a = "Hello, World!"
print(a.lower())        # returns "hello, world!"
```

The **upper**( ) method returns the string in upper case:

```
a = "Hello, World!"
print(a.upper())          # returns "HELLO, WORLD!"
```

The **replace**( ) method replaces a string with another string:

```
a = "Hello, World!"
print(a.replace("H", "J"))        # returns " Jello, World!"
```

The **split**( ) method splits the string into substrings if it finds instances of the separator:

```
a = "Hello, World!"
print(a.split(","))                  # returns ['Hello', ' World!']
```

String Format

- The **format**( ) method takes the passed arguments, formats them, and places them in the string where the placeholders {}.
- We can use index numbers {0} to be sure the arguments are placed in the correct placeholders

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."
print(myorder.format(quantity, itemno, price))
```

Output

I want to pay 49.95 dollars for 3 pieces of item 567.

Triple Quotes

- Python's triple quotes comes to the rescue by allowing strings to span multiple lines, including NEWLINEs, TABs, and any other special characters.
- The syntax for triple quotes consists of three consecutive **single or double** quotes.

```
a = " " "Hello,
        Python,
        Programming" " "
print(a)
```

```
a = ' ' ' Hello,
        Python,
        Programming ' ' '
print(a)
```

String Formatting Operator

- One of Python's coolest features is the string format operator %.
- This operator is unique to strings and makes up for the pack of having functions from C's printf() family.

```
print ("My name is %s and weight is %d kg!" % ('Zara', 21))
```

Output : My name is Zara and weight is 21 kg!

Sr.No.	Format Symbol & Conversion
1	%c character
2	%s string conversion via str() prior to formatting
3	%i signed decimal integer
4	%d signed decimal integer
5	%u unsigned decimal integer
6	%o octal integer
7	%x hexadecimal integer (lowercase letters)
8	%X hexadecimal integer (UPPERcase letters)
9	%e exponential notation (with lowercase 'e')
10	%E exponential notation (with UPPERcase 'E')
11	%f floating point real number
12	%g the shorter of %f and %e
13	%G the shorter of %f and %E

1. Program to check whether the string is a palindrome or not.

```
str=input("Enter the String")
l=len(str)
p=l-1
index=0
while (index<p):
 if(str[index]==str[p]):
   index=index+1
   p=p-1
 else:
   print ("String is not a palidrome" )
   break
else:
  print ("String is a Palidrome"  )
```

2. Program to count no of ‘p’ in the string pineapple.

```
word = 'pineapple'
count = 0
for letter in word:
  if letter == 'p':
     count = count + 1
print(count)
```

Assignment

1. Input a string “Green Revolution”. Write a script to print the string in reverse.
2. Consider the string str=”Global Warming”

   Write statements in python to implement the following

   a) To display the last four characters.

   b) To display the substring starting from index 4 and ending at index 8.

   c) To check whether string has alphanumeric characters or not.

   d) To trim the last four characters from the string.

   e) To trim the first four characters from the string.

   f) To display the starting index for the substring „Wa“.

   g) To change the case of the given string.

   h) To check if the string is in title case.

   i) To replace all the occurrences of letter „a“ in the string with „*“
3. Write a program to print the pyramid.

   1

   2 2

   3 3 3

   4 4 4 4

   5 5 5 5 5
4. What will be the output of the following statement? Also justify for answer.

```
>>> print 'I like Gita\'s pink colour dress'.
```

5. Give the output of the following statements

```
>>> str='Honesty is the best policy'
>>> str.replace('o','*')
```

6. Give the output of the following statements

```
>>> str='Hello World'
>>>str.istiltle()
```

7. Give the output of the following statements.

```
>>> str="Group Discussion"
>>> print str.lstrip("Gro")
```