MODULE 3 : PYTHON ASSIGNMENT

1)

```
if _name_ == '_main_':
   N = int(input())
   m=list()
   for i in range(N):
     method,*l=input().split()
     k=list(map(int,l))
     if len(k)==2:
       q=[k[0]]
       w=[k[1]]
     elif len(k)==1:
       q=[k[0]]
     if method =='insert':
       m.insert(q[0],w[0])
     elif method == 'append':
       m.append(q[0])
     elif  method == 'remove':
       m.remove(q[0])
     elif method =='print':
       print(m)
     elif method == 'reverse':
       m.reverse()

     elif method =='pop':

       m.pop()

     elif method == 'sort':

      m.sort()
```

2) PYTHON PROGRAM CALCULATOR

Example: Simple Calculator by Using Functions

```
# Program make a simple calculator


# This function adds two numbers

def add(x, y):

   return x + y


# This function subtracts two numbers

def subtract(x, y):

   return x - y


# This function multiplies two numbers

def multiply(x, y):

   return x * y
```

```
# This function divides two numbers
def divide(x, y):
    return x / y
print("Select operation.")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide"

while True:
    # take input from the user
    choice = input("Enter choice(1/2/3/4): ")

    # check if choice is one of the four options
    if choice in ('1', '2', '3', '4'):
        num1 = float(input("Enter first number: "))
        num2 = float(input("Enter second number: "))

        if choice == '1':
            print(num1, "+", num2, "=", add(num1, num2))

        elif choice == '2':
            print(num1, "-", num2, "=", subtract(num1, num2))

        elif choice == '3':
            print(num1, "*", num2, "=", multiply(num1, num2))

        elif choice == '4':
            print(num1, "/", num2, "=", divide(num1, num2))

        # check if user wants another calculation
        # break the while loop if answer is no
        next_calculation = input("Let's do next calculation? (yes/no): ")
        if next_calculation == "no":
          break

    else:
        print("Invalid Input")
```

3 PROGRAM TO CONCATENATE , REVERSE AND SLICE A STRING :

```
#include <stdio.h>
#include <stdlib.h>

int stringLength(char c[]) //because geany plays games with me
```

```
{
	int n = 0;
	char *s = &(c[0]);
	while(*s != '\0')
	{
		s = s + sizeof(char);
		n++;
	}
	return n;
}

int main()
{
	char string1[20]; //string entered by the user
	char string2[20];
	char revString1[20];// variable to store reversed string
	char revString2[20];
	int c, rc;       //counter and reverseCounter
	int cc1 = 0;          // concat counter
	int cc2;             //reverse concat counter
	printf("Enter a string: ");
	scanf("%s",string1);
```

```
printf("\nEnter another string: ");

scanf("%s",string2);

cc2 = stringLength(string2); //reverse concat counter

c = 0; //initialise counter with 0

rc = stringLength(string1) - 1; //rc = length of string entered


//Initialize a variable with double the length of original

// string entered by the user

char concat[2*stringLength(string1)];


//start the process of reversing


while(c < stringLength(string1) && rc >=0)

{

        revString1[c] = string1[rc]; //reversing

        revString2[c] = string2[rc];

        c++;

        rc--;

}


//start the process of concatenation
```

```
        while(cc1 < stringLength(string1) && cc2 > 0)

        {

                concat[cc1] = revString1[cc1]; //start with index 0 of the revString1


                //start with

                concat[cc1 + stringLength(string1)] = revString2[cc1];

                cc1++;

                cc2--;

        }

        printf("\nReversedString1: %s",revString1);

        printf("\nRversedString2: %s",revString2);

        printf("\nReversedString after concatanation with original string:
%s",concat);

        return 0;

}
```

4 PYTHON A POPULAR PROGRAMMING LANGUAGE:

Python is easy to learn It uses a simplified syntax with an emphasis on natural language, for a much easier learning curve for beginners. And, because Python is free to use and is supported by an extremely large ecosystem of libraries and packages, it's often the first-choice language for new developers.

9 Factors of Python Popularity

Why is Python so popular? What is it about Python that seems to capture the interest of developers, new and experienced alike? Here, we take a brief look at nine factors that have helped make Python one of the world’s leading programming languages.

1. Python is easy to learn

One of the largest hurdles for those who are interested in getting into coding is that programming languages really are their own languages; they have their own rules, syntax, grammatical structures, etc., and they often necessitate learning a completely new vocabulary.

But Python is different. More so than nearly any other programming language, Python reads and writes very similarly to standard English. It uses a simplified syntax with an emphasis on natural language, for a much easier learning curve for beginners. And, because Python is free to use and is supported by an extremely large ecosystem of libraries and packages, it’s often the first-choice language for new developers. These and other factors help demonstrate why Python is the best choice for those without proper coding experience.

2. Python has an active, supportive community

No programmer is an island; they depend on essential documentation and support so that when they encounter unexpected issues or new problems to solve, they have somewhere to go to find answers. Python has been around for over three decades, more than enough time for a dedicated user community to grow up around it. The Python community includes developers of all skill levels and provides easy access to documentation, guides, tutorials, and more.

At the same time, the Python community is extremely active. When developers are up against deadlines and in desperate need of help, they can work with the community to crowdsource fast, effective solutions.

3. Python is flexible

Python is often described as a general-purpose programming language. This means that unlike domain-specific languages which are designed only for certain application types, Python may be used to develop nearly any kind of application in any industry or field.

What is Python used for? Python has been used to great effect in web development, data analytics, machine learning, data science, data engineering, and even machine learning and artificial intelligence. Many top businesses and software companies depend on Python including Facebook, Google, Netflix, Instagram, and others. Supported by a range of frameworks and libraries, there’s essentially no coding job that Python can’t handle.

4. Python offers versatile web-development solutions

Although Python is an effective choice for many kinds of development projects, its usefulness in web development is worth specific recognition. Using available open-source libraries, Python developers can get their web applications up and running quickly and easily.

And while other languages, such as Java or .NET, might offer increased performance, the speed and developer experience provided by Python makes it an obvious choice for those who need a quick solution that they can depend on. At the same time, Python’s variety of available resources offers a unique opportunity to integrate other application types into websites.

5. Python is well suited to data science and analytics

Many of the factors that make python an attractive choice for beginners also set it apart as a reliable option for data-science and data-analysis. Python's ease of use, support, and flexibility have made it an essential tool for those who work with machine learning, cloud computing, and big data.

Python is particularly effective for analyzing and organizing data sets. In fact, for data science and analytics projects, Python is second only to R language in terms of popularity. Its out-of-the-box data analysis capabilities, combined with its growing ecosystem of data-focused frameworks, help ensure that Python remains a popular data-science programming solution.

6. Python is efficient, fast, and reliable

Occasionally, a developer that specializes in a different programming language might ask "Why is Python slow?" And yes, compared to some other languages, such as Java, C#, Go, JavaScript, or C++, Python often has a slightly slower execution speed. However, in today's world, development time is much more important than computer run time. And in terms of time-to-market, Python simply cannot be beaten.

Likewise, Python is efficient and reliable, allowing developers to create powerful applications with a minimum of effort. Completing coding projects is easy rather than time-consuming, and the results are able to stand toe to toe with applications created using more-demanding languages.

7. Python is widely used with IoT Technology

As wireless access becomes ever-more ubiquitous, the internet of things (IoT) continues to grow. These small, internet-connected devices often allow users to make small adjustments to their code, customizing their performance to fit specific needs. Many of these devices support either Python or Micropython (a scaled down version of the programming language designed for simpler devices).

As more and more devices become internet-connected, users are discovering that a working understanding of Python may be essential in fully utilizing the ever-expanding internet of things.

8. Python empowers custom automation

Programming complex technologies generally requires writing a significant amount of code. Unfortunately, even small, simple tasks can end up co-opting large amounts of available developer time. Python incorporates tools and modules to help automate these repetitive, time-consuming tasks, so that developers can focus their energies on other important issues.

Extended by its library of plugins, Python has become an automation standard across industries. In fact, even when working with other programming languages, developers will often write their automation scripts using Python.

9. Python is the academic language

Thanks to its growing dependability in the areas of data science, Python has become the go-to computing language in schools, colleges, and other places of learning. Simply put, those who pursue a formal education in computer science are extremely likely to be introduced to Python during the course of their learning and are even more likely to continue using Python throughout their career.

By teaching the next generation of programmers and developers how to get the most out of Python, schools are ensuring that Python remains a viable, popular option for years to come.

5 FRAMEWORKS USED WITH PYTHON :

Python is the go-to programming language for Data Science. Besides its inherent simplicity, what makes Python most appealing is that it is backed by a wide range of Python frameworks. Python frameworks offer a well-defined structure for app development. Since they can automate the implementation of some standard solutions, they not only reduce the development time significantly but also allow Developers to focus on the core application logic instead of routine elements. Long story short – they make the job of Developers much easier and make Python one of the best programming language.

6 WSGI:

The Web Server Gateway Interface (WSGI, pronounced whiskey or WIZ ghee) is a simple calling convention for web servers to forward requests to web applications or frameworks written in the Python programming language. The current version of WSGI, version 1.0.