SACP programs (**CLICK HERE for programs**)

Prescribed Text Book - Introduction to Python Programming (click here to open the textbook)

Sacp notes

Unit-1

What is compilation?

Compilation is the process of converting source code written in a high-level programming language into a lower-level language, such as machine code, that a computer can execute. This process is usually performed by a software program called a compiler.

The compiler checks the source code for the syntactical or structural errors, and if the source code is error-free, then it generates the object code.

What is interpreter?

An interpreter in programming is a computer program that executes instructions in a programming language without compiling them into a machine language program first. It translates high-level programming languages into machine code, line by line, and checks for errors.

Python compiled or interpreted?

Python is primarily an interpreted language , but it also has some aspects of a compiled language.

Interpreted Nature:

- Execution : Python code is typically executed by an interpreter, such as CPython, which reads the code line by line and executes it directly.
- Flexibility : This allows for more dynamic features, such as interactive testing and dynamic typing.

Compiled Aspects:

- Bytecode Compilation : Before executing, Python code is first compiled into an intermediate form called bytecode (.pyc files). This bytecode is then interpreted by the Python virtual machine (PVM). This step is usually invisible to the user.

- JIT Compilation : Some Python implementations, like PyPy, use Just-In-Time (JIT) compilation to improve performance by compiling parts of the code to machine code at runtime.

In summary, while Python is generally considered an interpreted language, it involves an initial bytecode compilation step, making it somewhat hybrid in nature.

The Python Virtual Machine (PVM) is an integral part of the Python interpreter. It is the engine that executes Python bytecode, which is the intermediate, platform-independent representation of your Python code.

How PVM Works:

1. Source Code : You write Python code in `.py` files.
2. Compilation to Bytecode : The Python interpreter first compiles your source code into an intermediate format called bytecode . This bytecode is stored in `.pyc` files, usually located in the `__pycache__` directory.
3. Execution by PVM : The PVM then interprets this bytecode, executing the instructions one by one on the host machine.

Key Points:

- The PVM abstracts away the details of the underlying hardware, allowing Python code to be portable across different platforms.
- While Python is often described as an interpreted language, the role of the PVM illustrates how the language actually combines both compilation (to bytecode) and interpretation (by the PVM).

In essence, the PVM is the core component that brings Python code to life, running the instructions encoded in bytecode on your machine.

Advantages and disadvantages of using interpreter:

Advantages:

1. **Interactivity**: The interpreter allows for interactive coding, enabling users to test snippets of code in real-time, which is great for debugging and experimentation.
2. **Portability**: Python code can run on any platform that has a Python interpreter, making it highly portable across different operating systems.
3. **Dynamic Typing**: Python is dynamically typed, meaning you don't need to declare variable types explicitly, which can speed up development.

4. **Rapid Prototyping**: The interpreted nature of Python allows for quick development and iteration, making it ideal for prototyping applications.

Disadvantages:

1. **Performance**: Interpreted languages like Python are generally slower than compiled languages (like C or C++) because the code is executed line by line.
2. **Memory Consumption**: Python can consume more memory than some other languages, which can be a concern for memory-intensive applications.
3. **Error Detection**: Errors in Python are often detected at runtime rather than at compile time, which can lead to runtime errors that are harder to debug.

Since Python programs are translated into this platform-independent bytecode, they can be run on any system with a Python interpreter for that platform.Python interpreters are available for multiple platforms (Windows, Linux, macOS, etc.), each tailored to translate the Python bytecode into the machine instructions for the specific operating system and hardware. This ensures the same Python code can run on multiple platforms without modification. **Thus python is known as Platform independent.**

Parts of Python Language:

Identifiers:

An identifier is a name given to a variable, function, class or module. Identifiers may be one or more characters in the following format:

• Identifiers can be a combination of letters in lowercase (a to z) or uppercase (A to Z) or digits (0 to 9) or an underscore (_). Names like myCountry, other_1 and good_ morning, all are valid examples. A Python identifier can begin with an alphabet (A – Z and a – z and _).

• An identifier cannot start with a digit but is allowed everywhere else. 1plus is invalid, but plus1 is perfectly fine.

● Keywords cannot be used as identifiers.

• One cannot use spaces and special symbols like !, @, #, $, % etc. as identifiers.

• Identifier can be of any length.

Keywords:

Keywords are a list of reserved words that have predefined meaning. Keywords are special

vocabulary and cannot be used by programmers as identifiers for variables, functions, constants or with any identifier name. Attempting to use a keyword as an identifier name will cause an error.

List of Keywords in Python

and as not

assert	finally	or	
break	for	pass	
class	from	nonlocal	
continue		global	raise
def	if	return	
del	import	try	
elif	in	while	
else	is	with	
except		lambda	yield
False	True	None	

Variables:

Follow the below-mentioned rules for creating legal variable names in Python.

• Variable names can consist of any number of letters, underscores and digits.

• Variable should not start with a number.

• Python Keywords are not allowed as variable names.

• Variable names are case-sensitive. For example, computer and Computer are different variables.

Assigning Values to Variables:

The general format for assigning values to variables is as follows:

variable_name = expression

Examples:

```
1. >>> number =100
2. >>> miles =1000.0
3. >>> name ="Python"
4. >>> number
100
5. >>> miles
1000.0
6. >>> name
'Python'
```

In ① integer type value is assigned to a variable number, in ② float type value has been assigned to variable miles and in ③ string type value is assigned to variable name. ④, ⑤ and
⑥ prints the value assigned to these variables.

In Python, not only the value of a variable may change during program execution but

also the type of data that is assigned. You can assign an integer value to a variable, use it as
an integer for a while and then assign a string to the variable. A new assignment overrides
any previous assignments. For example,

```
1. >>> century = 100
2. >>> century
100
3.>>> type(century)
Class <int>
>>> century = "hundred"
4. >>> century
'hundred'
>>> type(century)
Class <str>
```

As shown in the above examples, (first datatype of century is int and next it is string), The type of the variable is determined at runtime, based on the value you assign to it. Thus, Python is called dynamically typed.

And we need not specify the datatype of a variable before using it.

Advantages of dynamic typing:

- Flexibility in coding.
- Fewer lines of code since type declarations are unnecessary.

Python allows you to assign a single value to several variables simultaneously. For example,

```
>>> a = b = c =1
 >>> a
1
3. >>> b
1
4. >>> c
1
```

Operators:

Python supports a variety of operators that can be classified into different categories based on their functionality. Below is a breakdown of all the operators supported by Python:

1. **Arithmetic Operators**
These are used for mathematical operations:

Operator	Description	Example
`+`	Addition	`3 + 2 = 5`
`-`	Subtraction	`5 - 3 = 2`
`*`	Multiplication	`3 * 2 = 6`
`/`	Division (float)	`7 / 2 = 3.5`
`//`	Floor Division	`7 // 2 = 3`
`%`	Modulus (remainder)	`7 % 2 = 1`
``	Exponentiation	`2  3 = 8`

2. Comparison (Relational) Operators
These operators compare values and return a boolean result:

Operator	Description	Example
`==`	Equal to	`3 == 3`
`!=`	Not equal to	`3 != 2`
`>`	Greater than	`5 > 2`
`<`	Less than	`2 < 5`
`>=`	Greater than or equal	`5 >= 5`
`<=`	Less than or equal	`2 <= 5`

3. Logical Operators
These operators are used to perform logical operations:

Operator	Description	Example
`and`	Logical AND	`(True and False) = False`
`or`	Logical OR	`(True or False) = True`
`not`	Logical NOT	`not True = False`

4. Bitwise Operators
These operators perform operations on the binary representations of integers:

Operator	Description	Example
`&`	Bitwise AND	`5 & 3 = 1`
`|`	Bitwise OR	`5 | 3 = 7`
`^`	Bitwise XOR	`5 ^ 3 = 6`
`~`	Bitwise NOT (inversion)	`~5 = -6`

| `<<` | Left shift | `5 << 1 = 10` |
| `>>` | Right shift | `5 >> 1 = 2` |

5. Assignment Operators
These operators are used to assign values to variables, with optional operations:

Operator	Description	Example
`=`	Assign	`x = 5`
`+=`	Add and assign	`x += 2` (x = x + 2)
`-=`	Subtract and assign	`x -= 3` (x = x - 3)
`*=`	Multiply and assign	`x *= 2`
`/=`	Divide and assign (float)	`x /= 2`
`//=`	Floor divide and assign	`x //= 2`
`%=`	Modulus and assign	`x %= 2`
`**=`	Exponentiation and assign	`x **= 3`
`&=`	Bitwise AND and assign	`x &= 3`
`|=`	Bitwise OR and assign	`x |= 3`
`^=`	Bitwise XOR and assign	`x ^= 3`
`<<=`	Left shift and assign	`x <<= 1`
`>>=`	Right shift and assign	`x >>= 1`

6. Identity Operators
These operators are used to compare the memory locations of two objects:

Operator	Description	Example
`is`	Evaluates to `True` if both variables point to the same object	`x is y`
`is not`	Evaluates to `True` if both variables point to different objects	`x is not y`

7. Membership Operators
These operators test for membership in sequences like strings, lists, or tuples:

Operator	Description	Example
`in`	Evaluates to `True` if a variable is found in a sequence	``'a' in 'apple'``
`not in`	Evaluates to `True` if a variable is not found in a sequence	``'x' not in 'apple'``

8. Ternary (Conditional) Operator
The ternary operator is used for conditional expressions:

python code
x = a if condition else b

Example:
python code
max_num = a if a > b else b

9. Special Operators

9.1 Object Operators (`is`, `is not`):
These are used to check whether two objects are identical (i.e., whether they refer to the same object in memory):

```python
a = [1, 2, 3]
b = a
print(a is b)   True
```

9.2 Augmented Assignment Operators:
Python supports combined operators for modifying a variable and then assigning it:

Python code
x += 2   Equivalent to x = x + 2

These are the core operators supported by Python, enabling various operations across data types and programming paradigms.

Data Types

Data types specify the type of data like numbers and characters to be stored and manipulated within a program. Basic data types of Python are

- Numbers
- Boolean
- Strings
- Complex
- None

Numbers

Integers, floating point numbers and complex numbers fall under Python numbers category.
They are defined as int, float and complex class in Python.
Integers can be of any length; it is only limited by the memory available.
A floating point number is accurate up to 15 decimal places. Integer and floating points are separated by decimal points.
Eg: 1 is an integer, 1.0 is floating point number.

Complex numbers are written in the form, x + yj,where x is the real part and y is the imaginary part.

```
z = 3 + 4j
z1 = 3 + 4j
z2 = 1 + 2j
z_sum = z1 + z2  # Result: (4+6j)

z = 3 + 4j
print(z.real)  # Output: 3.0
print(z.imag)  # Output: 4.0

z = 3 + 4j
print(abs(z))        # Output: 5.0 (magnitude)
print(z.conjugate()) # Output: (3-4j)
```

None

None is another special data type in Python. None is frequently used to represent the absence of a value. For example,

1. >>> money = None

None value is assigned to variable money

Indentation

In Python, Programs get structured through indentation (FIGURE 2.3). Usually, we expect indentation from any program code, but in Python it is a requirement and not a matter of style. This principle makes the code look cleaner and easier to understand and read. Any statements written under another statement with the same indentation is interpreted to belong to the same code block. If there is a next statement with less indentation to the left, then it just means the end of the previous code block.

Comments

Comments are an important part of any program. A comment is a text that describes what the program or a particular part of the program is trying to do and is ignored by the Python interpreter. Comments are used to help you and other programmers understand, maintain, and debug the program. Python uses two types of comments: single-line comment and multiline comments.

Single Line Comment

In Python, use the hash (#) symbol to start writing a comment. Hash (#) symbol makes all text following it on the same line into a comment. For example,

```
#This is single line Python comment
```

Multiline Comments

Triple quotes are generally used for multiline strings. However, they can be used as a multiline comment as well.

For example,

```
'''This is
multiline comment
in Python using triple quotes'''
```

Reading Input

In Python, input() function is used to gather data from the user. The syntax for input function is,
variable_name = input([prompt])

prompt is a string written inside the parenthesis that is printed on the screen.
1. >>> person = input("What is your name?")
2. What is your name? mecs
3. >>> person
mecs

Even when the user inputs a number, it is treated as a string which should be casted or converted to number explicitly using appropriate type casting function.
>>> n=input('enter a number')
4
>>> type(n)
Class str
>>>n=int(n)
>>>type(n) # type function is used to check the datatype of the variable.
Class int

In the above example, n is converted into integer using int().converting a variable from one datatype to another is called type conversion or type casting.

A single statement to take an integer as input:
n=int(input()) or
n=int(input('enter a number'))

Print Output

The print() function allows a program to display text onto the console. The print function will print everything as strings and anything that is not already a string is automatically converted to its string representation. For example,
1. >>> print("Hello World!!")
Hello World!!
① prints the string Hello World!! onto the console. Notice that the string Hello World is enclosed within double quotes inside the print() function.

2. str.format() Method
Use str.format() method if you need to insert the value of a variable, expression or an object into another string and display it to the user as a single string. The format() method returns a new string with inserted values.
Examples:
name = "Alice"
age = 30

formatted_string = "My name is {} and I am {} years old.".format(name, age)
print(formatted_string)
Output: My name is Alice and I am 30 years old.

Using Positional Arguments:
formatted_string = "My name is {0} and I am {1} years old.".format(name, age)
print(formatted_string)
Output: My name is Alice and I am 30 years old.

Using Named Arguments:
formatted_string = "My name is {name} and I am {age} years old.".format(name="Alice", age=30)
print(formatted_string)
Output: My name is Alice and I am 30 years old.

Specifying Format (like float precision):
price = 49.99
formatted_string = "The price is {:.2f}".format(price)
print(formatted_string)
output:The price is 49.99
Program to Demonstrate the Positional Change of Indexes of Arguments
1. a = 10
2. b = 20
3. print("The values of a is {0} and b is {1}".format(a, b))
4. print("The values of b is {1} and a is {0}".format(a, b))
Output
The values of a is 10 and b is 20
The values of b is 20 and a is 10

>>> print("Give me {ball} ball".format(ball = "tennis"))
Give me tennis ball
f-strings
Formatted strings or f-strings were introduced in Python 3.6. A f-string is a string literal that is prefixed with “f”. These strings may contain replacement fields, which are expressions enclosed within curly braces {}. The expressions are replaced with their values.
Code to Demonstrate the Use of f-strings with print() Function
1. country = input("Which country do you live in?")
2. print(f"I live in {country}")

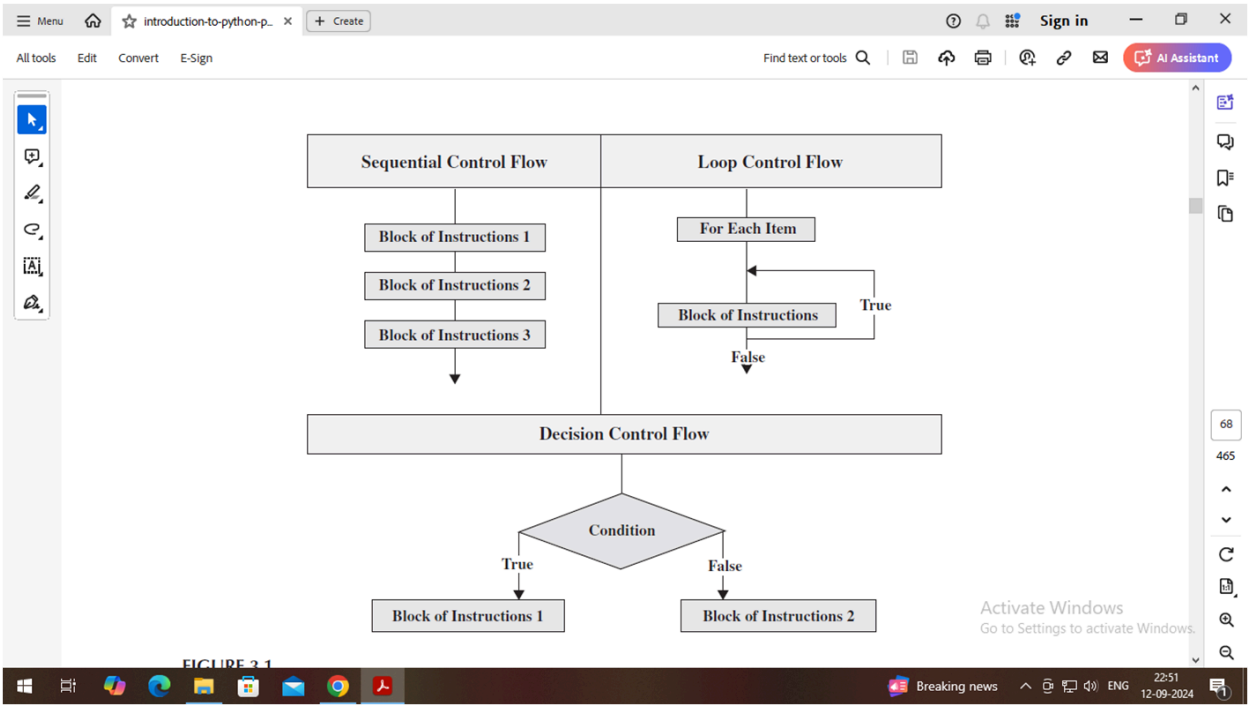
Output
Which country do you live in? India
I live in India

Practice:
Write Python Program to Convert the Given Number of Days to a Measure of Time Given in Years, Weeks and Days. For Example, 375 Days Is Equal to 1 Year, 1 Week and 3 Days (Ignore Leap Year).

The **float()** function returns a floating point number constructed from a number or string.
The **str()** function returns a string which is fairly human readable.
chr()-Convert an integer to a string of one character whose ASCII code is same as the integer using **chr()** function. The integer value should be in the range of 0–255.
The **ord()** function returns an integer representing Unicode code point for the given Unicode character.
The **hex()** Function Convert an integer number (of any size) to a lowercase hexadecimal string prefixed with "0x" .
The **oct()** Function Convert an integer number (of any size) to an octal string prefixed with "0o".
The operators **is and is not** are identity operators. Operator is evaluates to True if the values of operands on either side of the operator point to the same object and False otherwise.

Control Flow Statements

The control flow statements (FIGURE 3.1) in Python Programming Language are
1. Sequential Control Flow Statements: This refers to the line by line execution, in which the statements are executed sequentially, in the same order in which they appear in the program.
2. Decision Control Flow Statements: Depending on whether a condition is True or False, the decision structure may skip the execution of an entire block of statements or even execute one block of statements instead of other (if, if…else and if…elif…else).
3. Loop Control Flow Statements: This is a control structure that allows the execution of a block of statements multiple times until a loop termination condition is met (for loop and while loop). Loop Control Flow Statements are also called Repetition statements or Iteration statements.



The if Decision Control Flow Statement

The syntax for if statement is,

if Boolean_ Expression:

statement (s)

Colon should be present at the end

The expression in an if statement should be a Boolean expression. The if statement decides whether to run some particular statement or not depending upon the value of the Boolean expression.

If the Boolean expression evaluates to True then statements in the if block will be executed; otherwise the result is False then none of the statements are executed.

For example,

1. >>> if 20 > 10:
2. … print(f"20 is greater than 10")

Output

20 is greater than 10

<u>Program Reads a Number and Prints a Message If It Is Positive</u>

1. number = int(input("Enter a number"))
2. if number >= 0:
3. print(f"The number entered by the user is a positive number")

Output
Enter a number 67
The number entered by the user is a positive number

The if…else Decision Control Flow Statement:

Statements in the if block are executed if the Boolean_Expression is True. Use the optional else block to execute statements if the Boolean_Expression is False.
The if…else statement allows for a two-way decision.
The syntax for if…else statement is,

```
if Boolean_Expression:
        statement_1
else:
        statement_2
```

Program to Find If a Given Number Is Odd or Even

```
1. number = int(input("Enter a number"))
2. if number % 2 == 0:
3.         print(f"{number} is Even number")
4. else:
5.         print(f"{number} is Odd number")
```

Output:
Enter a number: 45
45 is Odd number

The if…elif…else Decision Control Statement

The if…elif…else is also called as multi-way decision control statement. When you need to choose from several possible alternatives, then an elif statement is used along with an if statement. The keyword 'elif' is short for 'else if' and is useful to avoid excessive indentation. The else statement must always come last, and will again act as the default action.
The syntax for if…elif…else statement is,

```
if Boolean_Expression_1:
        statement_1
elif Boolean_Expression_2:
        statement_2
elif Boolean_Expression_3:
        statement_3
:
:
:
else:
        statement_last
```

This if…elif…else decision control statement is executed as follows:
• In the case of multiple Boolean expressions, only the first logical Boolean expression

which evaluates to True will be executed.
• If Boolean_Expression_1 is True, then statement_1 is executed.
• If Boolean_Expression_1 is False and Boolean_Expression_2 is True, then statement_
2 is executed.
If Boolean_Expression_1 and Boolean_Expression_2 are False and Boolean_
Expression_3 is True, then statement_3 is executed and so on.
• If none of the Boolean_Expression is True, then statement_last is executed.

Write a Program to Prompt for a Score between 0.0 and 1.0. If the Score Is Out of Range, Print an Error. If the Score Is between 0.0 and 1.0, Print a Grade Using the Following Table

Score Grade
>= 0.9 A
>= 0.8 B
>= 0.7 C
>= 0.6 D
< 0.6 F

```
score = float(input("Enter your score"))
2. if score < 0 or score > 1:
3.          print('Wrong Input')
4. elif score >= 0.9:
5.          print('Your Grade is "A" ')
6. elif score >= 0.8:
7.          print('Your Grade is "B" ')
8. elif score >= 0.7:
9.          print('Your Grade is "C" ')
10. elif score >= 0.6:
11.         print('Your Grade is "D" ')
12. else:
13.         print('Your Grade is "F" ')
```

Output

```
Enter your score0.92
Your Grade is "A"
```

Program to Display the Cost of Each Type of Fruit

```
1. fruit_type = input("Enter the Fruit Type:")
2. if fruit_type == "Oranges":
3.          print('Oranges are $0.59 a pound')
4. elif fruit_type == "Apples":
5.          print('Apples are $0.32 a pound')
6. elif fruit_type == "Bananas":
7.          print('Bananas are $0.48 a pound')
8. elif fruit_type == "Cherries":
9.          print('Cherries are $3.00 a pound')
10. else:
11.         print(f'Sorry, we are out of {fruit_type}')
```

Output
Enter the Fruit Type: Cherries
Cherries are $3.00 a pound

Nested if Statement:
In some situations, you have to place an if statement inside another statement. An if statement that contains another if statement either in its if block or else block is called a Nested if statement.
The syntax of the nested if statement is,

```
if Boolean_Expression_1:
        if Boolean_Expression_2:
                Statement_1
        else:
                statement_2
else:
        statement_3
```

Program to Check If a Given Year Is a Leap Year

```
1. year = int(input('Enter a year'))
2. if year % 4 == 0:
3.         if year % 100 == 0:
4.                 if year % 400 == 0:
5.                         print(f'{year} is a Leap Year')
6.                 else:
7.                         print(f'{year} is not a Leap Year')
8.         else:
9.                 print(f'{year} is a Leap Year')
10. else:
11. print(f'{year} is not a Leap Year')
```

Output
Enter a year 2014
2014 is not a Leap Year

The while Loop:

The syntax for while loop is,

```
while Boolean_Expression:
        statement(s)
```

With a while statement,
the first thing that happens is that the Boolean expression(condition) is evaluated before the statements in the while loop block is executed. If the Boolean expression evaluates to False, then the statements in the while loop block are never executed. If the Boolean expression evaluates to True, then the while loop block is executed. After each iteration of the loop block, the Boolean expression is again checked, and if it is True, the loop is iterated again. Each repetition of the loop block is called an iteration of the loop. This process continues until the Boolean expression evaluates to False and at this point the while statement exits. Execution then continues with the first statement after the while loop.

Write Python Program to Display First 10 Numbers Using while Loop Starting from 0

```
1. i = 0
2. while i < 10:
3.         print("Current value of i is”,i)
4.          i = i + 1
```

Output

```
Current value of i is 0
Current value of i is 1
Current value of i is 2
Current value of i is 3
Current value of i is 4
Current value of i is 5
Current value of i is 6
Current value of i is 7
Current value of i is 8
Current value of i is 9
```

Write a Program to Find the Average of n Natural Numbers Where n Is the Input from the User

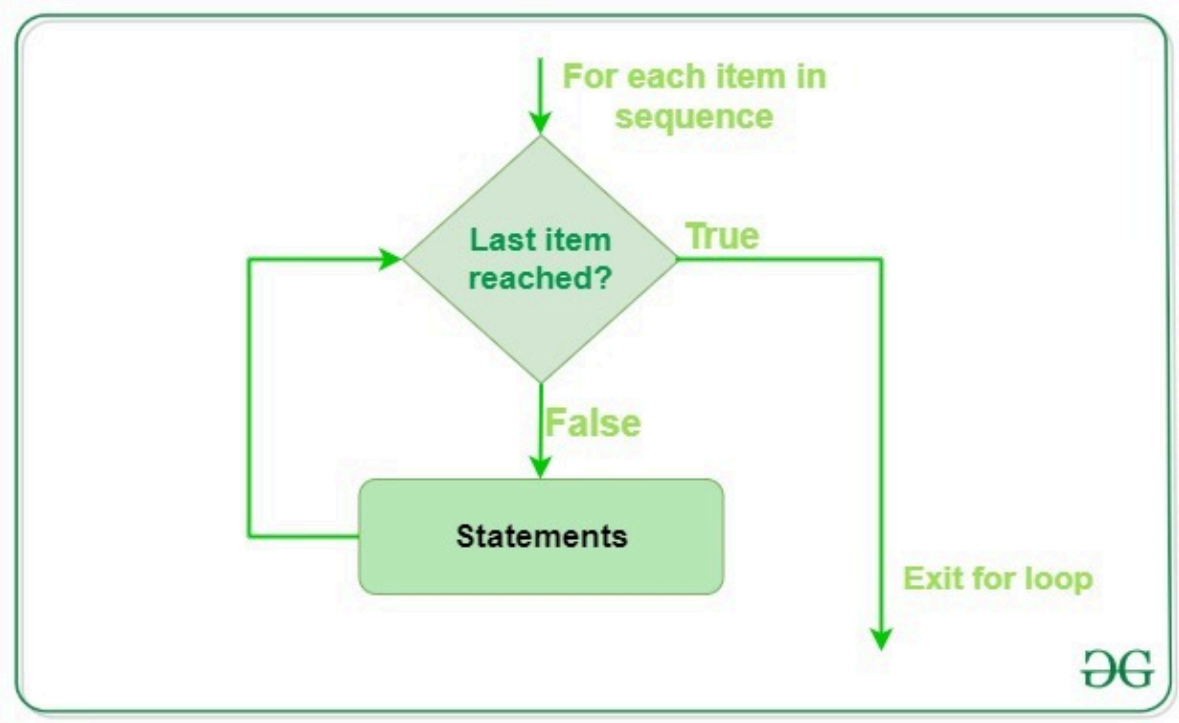
```
1. number = int(input("Enter a number up to which you want to find the average"))
2. i = 0
3. sum = 0
4. count = 0
5. while i < number:
6.          i = i + 1
7.         sum = sum + i
8.         count = count + 1
9. average = sum/count
10. print(f"The average of {number} natural numbers is {average}")
```

Output

```
Enter a number up to which you want to find the average 5
The average of 5 natural numbers is 3.0
```

For Loops:

The **For Loops in Python** are a special type of loop statement that is used for sequential traversal.



the for loop is used to iterate over a sequence (such as a list, tuple, string, or dictionary) or any iterable object. The basic syntax of the for loop is:

```
for var in iterable:
    # statements
    pass
```

Python For Loop with String

This code uses a for loop to iterate over a string and print each character on a new line. The loop assigns each character to the variable i and continues until all characters in the string have been processed.

```
# Iterating over a String
print("String Iteration")

s = "MECS"
for i in s:
    print(i)
```

Output:

String Iteration

M

E

C

S

Python for loop with Range

This code uses a Python for loop with index in conjunction with the range() function to generate a sequence of numbers starting from 0, up to (but not including) 10, and with a step size of 2. For each number in the sequence, the loop prints its value using the print() function. The output will show the numbers 0, 2, 4, 6, and 8.

```
for i in range(0, 10, 2):
    print(i)
```

Output :

0

2

4

6

8

Nested For Loops in Python

This code uses nested for loops to iterate over two ranges of numbers (1 to 3 inclusive) and prints the value of i and j for each combination of the two loops. The inner loop is executed for each value of i in the outer loop. The output of this code will print the numbers from 1 to 3 three times, as each value of i is combined with each value of j.

```
for i in range(1, 4):
    for j in range(1, 4):
        print(i, j)
```

Output :

1 1

1 2

1 3

2 1

2 2

2 3

3 1

3 2

3 3

SACP programs

Basics:

1. Write a program to use variables and assign different types of data to those and display the datatype of those variables.
2. To find the square of a number
3. To find the square of a given number
4. To find the area of a room
5. To find the area of a circle for the given radius.
6. Take three numbers as input and perform different arithmetic operations on them.
7. A newly opened store is offering a 15% discount on the MRP value of any product.

   Write a program that takes the value of a product in rupees as input and outputs the price after discount.

   Input Format:

   The first line contains a user input number

   Output Format:

   Print the value after discount as the output

   Example:

   Input:

   100

   Output:

   85.0

8. Write and check what are the operations supported by strings
9. Take two strings as input and perform different string operations on them.
10. To take the details of a student(name, rollnumber,eamcet rank, cgpa) as input and print them

11. a)Let s=”mecs”, print “mecsmecsmecsmecs” without loops. b) take a string as input and print it given number of times separated by$
12. Write a command to Swap two variables, two strings.
13. Write a program to convert the temperature in degree centigrade to Fahrenheit.
14. Write a program to find the circumference and area of a circle with a given radius.

Programs on if - elif - else

1. write a program to check whether a person with given age is eligible to vote or not
2. Write a program to check whether the given year is leap year or not.
3. write a program to check if given number is even or odd
4. write a program to display ‘P’ if given marks are >=40 , else ‘F’
5. Write a program that takes 3 integers as input and prints the difference between the largest and smallest value among the three.

   Input Format:

   Single line of input contains three numbers

   Output Format:

   Print the value after subtracting the smallest number from the largest

   Example:

   Input:

   5 7 9

   Output:

   4

6. Write a program that takes 3 integers as input and checks whether they can form the sides of a right angled triangle or not. Print YES if they can form a right angled triangle. NO, otherwise.

   Input Format:

   Single line of input contains three numbers

   Output Format:

   Print YES or NO

Example:

Input:

5 4 3

Output:

YES

Example:

Input:

10 20 30

Output:

NO

7. Write a program that takes 3 integers as input and checks whether they can form the sides of a triangle or not. Print YES if they can form a triangle. NO, otherwise.

   Input Format:

   Single line of input contains three numbers

   Output Format:

   Print YES or NO

   Example:

   Input:

   5 7 10

   Output:

   YES

   Example:

   Input:

   7 8 16

Output:

NO

8. Accept the electric units from user and calculate the bill according to the following:

   First 100 units: Free

   Next 200 units : Rs 2 per unit

   above 300 units: Rs 5 per unit

   If number of units are 500 then total bill is : 0+ 400+ 1000 = 1400

Programs using loops: (programs 8-12 are syllabus book programs)

1. Write a program to print first 10 odd numbers

   o/p: 1 3 5 7 9 11 13 15 17 19

2. Write a program to print first ten even numbers in the reverse order
3. Write a program to display the sum and product of the digits of a given number.
4. **Print numbers divisible by 3 or 5 from 1 to 20 using a for loop**:
5. Write a program to **Print characters of a string using a for loop**
6. **write a program to print numbers from 1 to 50 , except multiples of 5.(using continue statement in the loop)**
7. Write a program to check whether the given number is prime or not.
8. Write a program to find the average of 10 numbers using while loop.
9. Write a program to display the given integer in a reverse manner.
10. Write a program to find the sum of the digits of an integer using a while loop.
11. Write a program to display all integers within the range 100-200 whose sum of digits is an even number.
12. Write a program to find the roots of a quadratic equation and display the nature of roots.
13. Write a program to find the factorial of a number using loop.
14. Write a program to find the Nth term in a Fibonacci series using loop.
15. Write a program to Print the multiples of 5 between given a and b
16. Write programs to print the following patterns:

    1 1 1 1 1

2 2 2 2 2

3 3 3 3 3

4 4 4 4 4

5 5 5 5 5

---

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

1

22

333

4444

55555

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

---

1 1 1 1 1

1 0 0 0 1

1 0 0 0 1

1 0 0 0 1

1 1 1 1 1

Exception handling:

1. Write a program to read two numbers a, b and printa/b , display an error message if b is entered as zero
2. write a program to read an integer and print it, print an error message if it is not a number
3. Write a program to read a number and display it , run the program till a number is entered
4. write a program to allow the user to enter the correct password , where maximum number of attempts allowed are only 3
5. Write a program to Print the multiples of 5 between given a and b, ensure that a<b

Programs using lists:

1. Write a program to read a list of elements and display the even numbers in the list.
2. Write a program to read list of elements using a for loop and display multiples of 5 in the list
3. Write a program to find the difference between largest and smallest in the list and print sum and average of elements in the list
4. Create two lists with numbers. Write a program to create two lists where first list contains all the numbers from both lists which are >8 and second list contains all the numbers from both lists which are<=8
5. Given a list of integers and a value k, print the number in the list that appears exactly k times(it is guaranteed that there's exactly one integer that appears k times)
6. Given a list of integers nad a value k, print the kth largest element in the list.(all the elements in the list are guaranteed to be distinct)
7. Given a list of n integers , print a new list such that every element in the new list represents the cumulative sum of all the elements until that position.

8. Write a program that prints the maximum value of the second half of the list
9. write a program that create a list of numbers 1-100 that are either divisible by 5 or 6
10. Write a program to create a list and display the sum of list members(syllabus book program)
11. Write a program to find the largest number in a list without using built-in function.(syllabus book program)

Note: practice indexing and slicing in the list

2d lists and functions:

1. write a program to read and print the matrix elements of 3*3
2. Write a program to add two matrices of n*n using functions
3. write a program to multiply two matrices and print the result.
4. Write a program to print the given list of elements in the required dimensions

   Input:

   First line: N (No.of elements)

   second line: 1 2 3 4 5 6 7 8 9 10

   Third line: dimensions: 5*2

   Output:

   1 2

   3 4

   5 6

   7 8

   9 10

   Input2:

   First line: N (No.of elements)

   second line: 1 2 3 4 5 6 7 8 9 10

Third line: dimensions: 6*2

Output: not possible

Strings:

1. Practice string operations and slicing
2. Write a program to convert the given time from 24 hour format to 12 hour format and vice versa
3. Write a program to check whether the given string is a palindrome or not
4. **Break the Secret**

   You are hired by a secret service agency. Your job is to decode messages. You figure out the algorithm used to encrypt messages and it turns out to be as follows:

   The message will consist of only uppercase alphabets. The positional values are assigned to the characters as A=0, B=1, ..., Y=24 and Z=25. In the original message, a character at position i is replaced by a character using the shift formula (i+3)%26.

   For example A will be replaced by D, B by E an so on. After replacement, the string is reversed and sent as a message.

   You are required to decode this message through your code.(Assuming No Space in the message)

   Input Format

   A single line of the input contains the encrypted message

   Output Format

   Print a string containing the original message after decoding

   Example:

   Input

   HBEGRRJ

   Output:

   GOODBYE

5. Write a program to print the characters which are common in two strings

6. Write a program to count the length of the string without using built in function.
7. Write a python program to that accepts a sentence and calculates the number of words, digits, uppercase and lowercase letters
8. Write a program to count the number of vowels in the given string.
9. Write a python program to convert uppercase to lowercase and viceversa.

Recursion:

1. Write a program to find the factorial of the given number using recursive method.
2. write a program to find the power of a,b using recursive method.
3. write a program to find the n th fibonacci number using recursion
4. Write a recursive program to find the gcd of two numbers
5. Write a function that prompts the user to enter five numbers, then invoke a function to find the GCD of these numbers.

Dictionary:

1. Write Python Program to Generate a Dictionary That Contains (i: i*i) Such that i Is a Number Ranging from 1 to n.

   Display the value for given key

   Display the key for given value

   print sum of key-value # pairs in dictionary

2. Write a Python Program to create a dictionary in which the keys are the characters and values are their frequencies in the given string. Display the dictionary in the sorted order.
3. Write a function named addfruit, which is passed with a set of fruit names and their prices and returns a dictionary containing the entered information and raises a ValueError exception if the fruit is already present.

   Display the price of the given fruit.

   Display the fruits below the given price.

4. Write a program to create a dictionary containing the student information(for 5 students) : name, roll no, marks in 4 subjects. Display the average marks of a given student.

   Change value of a key in a nested dictionary

Display the maximum and minimum marks along with the subject info of a given student.

Using the above dictionary do the following:

a) Create a dictionary final which has names and total marks and also find the topper.

5. Create a dictionary which contains subject name as key and marks as values.

    a) Change the marks to True/False : true if marks>=40 else False
    b) Display if the student is pass or fail

    HINT: use all, any functions

Eg: d1= {'c':50,'java':60,'python':70}

Change this to d1={'c':TRUE,'java':TRUE,'python':TRUE}

O/P: Pass

Eg 2: d1= {'c':40,'java':60,'python':70}

Change this to d1={'c':False,'java':TRUE,'python':TRUE}

O/P: Fail

Hint: use any, all functions

5. Write a program that has the dictionary of your friends' names as keys and phone numbers as its values. Print the dictionary in a sorted order. Prompt the user to enter the name and check if it is present in the dictionary. If the name is not present, then enter the details in the dictionary. Allow multiple phone numbers for a student.

   Delete the item with a given key.

   Change the phone number of a given name to the entered phone number.

6. Write a program to create a dictionary to convert values from meters to centimeters(1m-10m).(write inline coding)

   Del the item with 5m as key.

Delete all the items.

Delete the entire dictionary.

Sets :

Write the programs to do the following:

1. You have a list of items (like product IDs or user names) that may contain duplicates, and you want only the unique values. Prepare it by converting the list to a set. product_ids = [1, 2, 3, 2, 4, 1, 5]
2. In a social network, find mutual friends between two users.
   alice_friends = {"John", "Mike", "Sarah"} bob_friends = {"Sarah", "Mike", "Anna"}
3. Check if a set of keywords are all present in a given text.
   text = "Python is a popular programming language for data science" keywords = {"Python", "data", "science"}
4. wtsappgrp1={‘raj’,’ram’,’rana’,’radha’}
   wtsappgrp2={‘’ram’,’radha’,’john’,’jack’}
   wtsappgrp3={‘jill’,’john’,’raj’,’ram’}
   Find the common groups for raj, ram
   John joined in how many groups?
5. Combine tags from different sources, like tags added by users and automated tags, to get the complete set of tags for an article. user_tags = {"python", "tutorial", "coding"} auto_tags = {"programming", "tutorial"}
6. You have a list of all login IPs and a list of known suspicious IPs, and you want to find which IPs in your list are not suspicious.
   all_ips = {"192.168.1.1", "192.168.1.2", "192.168.1.3"}
   suspicious_ips = {"192.168.1.2"}
7. A user has two shopping carts, and find items that are in only one of the carts, not both. cart1 = {"apple", "banana", "cherry"} cart2 = {"banana", "date", "cherry"}

Files:

(create a file and add some text)
Write programs to do the following:

1. To display the contents of the file using read()
2. To display the contents of the file line by line
3. To display the words of the file
4. To count the number of lines, words, vowels in the file
5. To copy the contents of the file to another file
6. To display each word in the reverse.
7. To capitalize each word in the file.
8. To read 10 bytes from 5th byte and print
9. To add the text " this is new text" to the file
10. To add the text " new text " at the beginning of the file.

Pandas:

1. Create a pandas series with the values [1,3,5,7,9]
2. Create a numpy array with even numbers from 1 to 15, and create a series from this.
3. Create a series with 5 values which are the prices of 5 items. Change the index as ['milk','bread','chocolate','pen','watch']
    a) Display the values from 1 to 3
    b) Display the last value
    c) Display the price of bread, pen,milk (in the given order)
4. Find the min,max,sum of the above series
5. Create a series of temperatures of 5 cities. Make the index as names of the cities. Find the city a) where minimum temperature is observed b) where maximum temperature is observed
6. Create a series of cities : hyderabad, chennai, calicut,mumbai, mangolore
    a) Display the series in uppercase
    b) Display the cities that starts with 'c'
    c) Display the cities that starts with 'm'

d) Find if letter ‘i’ is present in the names of the cities

e) Find the position of ‘i’ if it is present in the names of the cities

7. Create the pandas data frame with the following data

a) Create two lists roll, theory_marks

b) From the above two lists, create dataframe

c) Display statistics of the data

Roll number	Theory Marks
61	85
62	86
63	96
64	95
65	93
66	85
67	80

8. Create the data frame with the following data

i) Create a dictionary with the data

ii) From the above dictionary, create dataframe

iii) Display the marks which are >60

iv) Display the roll numbers who got less than 50

Roll number	Lab Marks
68	65
69	66
73	46
74	55

65	73
66	75
67	70

9. To the above dataframe, add the column Theory marks (with values in Q7)

i) display the theory marks only

ii) display the lab marks only

iii) add the column “total” with total marks(theory marks+lab marks) as the values

10. Link to data file:

https://drive.google.com/file/d/1nqO-1JXVTzu9H0U4TaEar-1TgMX9sdH_/view?usp=drive_link

a) Download the data from the above link and create a dataframe
b) Identify the missing values in each column(count)
c) Display the total missing values in the data frame
d) Use different methods for filling missing values in different columns
e) Visualise the data in different charts.

Try the following:

a)
```
a={i:i*2 for i in range(5)}
```

```
b={i:i*3 for i in range(5)}

print(a,b)
```

```
a.update(b)

print(a,b)
```

Practice:

1.String is _____

- Collection of numbers
- Collection of characters
- Collection of program
- Collection of codes

2.Which of the following is not a valid variable name in Python?

- _sum
- first_sum
- sum2
- 2sum

3.Which symbol is used for writing comment in python?

- #
- !
- ^
- *

4.Is Python a case sensitive programming language?

- Yes
- No
- Maybe
- Not Sure

5.

6/2 is ____

- 3
- 3.0

1

2

6.3//2 is ____

1.5

1.6

1

0

7. What is the answer to this expression, 29 % 3 ?

4

3

2

1

8.Which one of these is floor division?

//

/

|

!

9. In the equation x + y, x and y are_____

Operands

Operators

Alphabets

Equation

10. What will be the output of the following code?

```
a1=20
a2=30
avg=(a1+a2)/2
print(avg)
```

20

30.0

25

25.0

11. What will be the output of the following code?

```
print(2**2**2 )
```

2

4

8

16

12. Which one of the following is a valid Python if statement ?

if (a >= 2)

if a>=2 :

if (a => 22)

if a >= 22

13. What does the following code print?

```
if 2 + 3 == 10:
   print("TRUE")
else:
   print("FALSE")
print("TRUE")
```

a) False

True

b) True

False

c) True

d) False

14. What keyword would you use to add an alternative condition to an if statement?

a)elif

b)elseif

c)else if

d) ifelse