```
#get marks from user and print the total

#variable with string - f string, + , { }
# print -> Hello Vivek! You are 18 years old. You got cash price of $999.99

name = 'Vivek'
print('Hi {name}')
print(f'Hi {name}')

name = "KIRUTHIKA"
print(name.isupper())
print(name.islower())
print(name.isdecimal())

word = "Women Empowerment"
print(word.upper().isupper())
print(word.index("w"))
print(word.index("erme"))

print(word.find("w"))
print(word.find("erme"))

print(word.index("z")) # error
print(word.find("z"))  # find

print( 5 * 6 + 11)
print( 5 * (6 + 11))
print( 13 % 5) #3
print ( 13 // 5) #2

my_number = -101
print(abs(my_number))
print(pow(2,4)) #2^4 = 2*2*2*2
print(max(250,3000))
print(min(250,3000))
print(round(6.4))
print(round(6.5)) #6
print(round(6.6))
print(round(6.9)) #7

#math - module
from math import *
print(round(3.7))
print(floor(3.7)) # 3
print(round(3.2))
print(ceil(3.2)) # 4
print(sqrt(121)) # 11
```

```
numbers = [23, 67, 89, 50]
```

```
print(numbers[-3:-1]) #[50,89,67]
print(numbers[-4:-1])
print(numbers[-1:-4])

print(numbers[-5:-1]) # error
print(numbers[-5])
```

```
a =10
a = False
print (a)
```

print(type(a)

# variable a will take the last value assigned to it

```
# Repetition - To print multiple times
#how will you print in next line? Homework
print(3 * 'vivek\n')
```

Notes

1. What is a programming language?

A **programming language** is a set of rules and syntax used to write instructions that a computer can understand and execute.

2. Why do we need programming language?

To interact with computer which can understand only 1’s & 0’s

3.Types of programming language:

High-level languages (e.g., Python, Java, JavaScript): Easier to write and understand, closer to human language.

Low-level languages (e.g., Assembly, C): Closer to machine code, harder to write but more control over hardware.

4. What is a variable?

A variable is a named storage location in memory that can hold a value, and that value can change during the program's execution

4. What is datatype in programming?

A **data type** in programming tells the computer **what kind of data** a variable is storing, like whether it's a number, text, or something else.

5. What is typecasting?

Typecasting (also called **type conversion**) is the process of **changing the data type** of a value or variable from one type to another.

6. What is programming paradigm?

- A **programming paradigm** is a style or way of programming.

Procedural Programming

- Step-by-step instructions (like a recipe).
- Organizes code into functions.
- Focused on what to do and how to do it.

Imperative Programming

- Tells the computer how to do things step-by-step.
- Overlaps a lot with procedural programming.

Object-Oriented Programming (OOP)

- Organizes code into classes and objects.
- Great for modeling real-world things.
- Uses concepts like inheritance, encapsulation, and polymorphism.

Functional Programming

- Focuses on functions, immutability, and pure functions (no side effects).
- Uses higher order functions like map(), filter(), lambda, and reduce().

6. What is Python programming language?

- **Python** is a high-level, interpreted programming language
- Python is a **multi-paradigm language**, which means you can write code in more than one style. It **supports** Procedural programming, object oriented language, functional programming
- It supports primitive data types like int, float,boolean,string to more complex data types like lists, set,tuples and dictionary
- A **dynamically typed language** is one where **you don't have to declare the type of a variable** when you write your code. The **type is determined at runtime**, meaning **while the program is running**, not before.

7. What is dynamically typed language?

In dynamically typed languages, **the type of a variable is determined at runtime**, meaning it's not explicitly declared and can change during execution.

8. What is meant by platform dependent language?

A platform-dependent language is one where the **compiled code is specific to a particular operating system or hardware platform.**

9. What is meant by platform independent language?

A platform-independent programming language is one that allows you to write code that can run on different operating systems and hardware architectures without needing to make changes to the code itself.

When you write a Java/Python program:

1. Java/Python code is compiled into bytecode first

2. This bytecode is executed by the Java Virtual Machine (JVM)/Python Virtual Machine.

3. The JVM/PVM is available for many platforms (Windows, macOS, Linux, etc.).

4. So, as long as the JVM/PVM is present, the bytecode can run — regardless of the underlying operating system.

In contrast, Languages like **C or C++** are usually compiled directly to machine code for a specific operating system and processor. This means they are **platform dependent** — you'd need to recompile your code for different platforms.

10. What is PVM (Python Virtual Machine)?

The Python Virtual Machine (PVM) is the **part of the Python interpreter** which takes the **bytecode** from compiler as input and runs/executes it line by line.

[Python is an **interpreted language** — the PVM interprets bytecode, not raw Python source code directly.]

11. What's the architecture of Python Virtual Machine?

- Before execution, Python source code is compiled into bytecode. This bytecode is a platform-independent.
- The heart of the Python VM is the **interpreter loop**. It fetches bytecode instructions, decodes them, and executes them line by line (sequentially).
- The Python VM maintains a **Python object model** to represent data types, such as integers, strings, lists, and custom objects. It manages the creation, manipulation, and destruction of these objects during execution.
- The Python VM **handles memory allocation and garbage collection** to manage the memory used by Python objects dynamically.

12. What is an interpreter?

Interpreter is a computer program that converts each high-level program statement into machine code

13. What is compiler?

A compiler is a special program that translates a programming language's source code into machine code, bytecode or another programming language.