PYTHON

**What is python? :-**

**Python** is a high-level, interpreted programming language known for its simple syntax and readability, making it ideal for beginners. It is widely used in web development, data science, automation, artificial intelligence, and more.

**What is a compiler? :-**

A **compiler** is a tool that changes your whole program into a language the computer can understand.

**FIRST PROGRAM: -**

**INPUT: -**

**A screen shot of a computer

AI-generated content may be incorrect.**

**OUTPUT: -**

**A black screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**This program gives the computer three simple instructions:**

1. **The first instruction tells the computer to show the message "HELLO WORLD" on the screen.**
   * **The words HELLO WORLD are written inside double inverted commas (" ") because they are a text message, also called a string.**
   * **This tells the computer: “This is just text, not a number or code.”**
2. **The second instruction tells the computer to show the number 14.**
   * **Since it's a number, it does not need any commas or quotation marks. The computer shows it as it is.**
3. **The third instruction tells the computer to multiply 7 by 2.**
   * **The computer solves this math problem and shows the result, which is 14.**
   * **This is a calculation, so it also doesn’t need quotation marks.**

**💡 Note on Inverted Commas:**

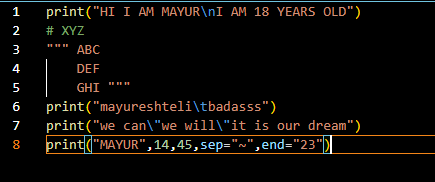
* **Strings (text) can be written in double inverted commas (" ") *or* single inverted commas (' ').**
* **For example: "HELLO" and 'HELLO' both mean the same thing in Python.**
* **It’s up to you which one you use, but they must match (start and end with the same type).**

**✅ In Simple Words:**

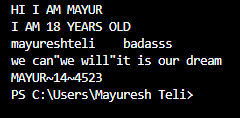
* **Use inverted commas when writing text.**
* **Don’t use them for numbers or math.**
* **You can use either double (" ") or single (' ') inverted commas for text.**

**COMMENTS, ESCAPE SEQUENCE,PRINT STATEMENT :-**

**INPUT:-**

****

**OUTPUT:-**

****

**EXPLANATION:-**

**🔹 First Line Explanation**

**A sentence is printed that includes the phrase “HI I AM MAYUR” followed by a newline escape sequence. The newline causes the next part, “I AM 18 YEARS OLD,” to appear on the next line.**

**Example Output:  
HI I AM MAYUR  
I AM 18 YEARS OLD**

**🔹 Comment Line Explanation**

**A line starts with a hash symbol, marking it as a comment. This line is ignored by Python during execution and does not appear in the output. It is used by the programmer for notes or explanations.**

**Example Output:  
(no output; the comment is not shown)**

**🔹 Triple Quoted String Explanation**

**A multi-line string spans three lines: “ABC”, “DEF”, and “GHI”. However, since this string is not printed or used in any way, it does not affect the output.**

**Example Output:  
(no output; unused string)**

**🔹 Tab Escape Explanation**

**A line prints the word “mayureshteli”, then uses a tab escape sequence to insert a wide horizontal space before printing the word “badass”.**

**Example Output:  
mayureshteli   badass  
(the spacing between the words is a tab space)**

**🔹 Escaped Double Quotes Explanation**

**A sentence includes double-quote characters around the phrase “we will”. Since double quotes normally define string boundaries, an escape character is used before each quote so they appear as part of the text.**

**Example Output:  
we can"we will"it is our dream**

**🔹 Custom Separator and End Explanation**

**A line prints the word “MAYUR” followed by the numbers 14 and 45. Instead of default spaces, a tilde symbol is used to separate them. The line ends not with a line break but with the number 23.**

**Example Output:  
MAYUR~14~4523**

**VARIABLES & DATA TYPE: -**

**INPUT: -**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**OUTPUT: -**

**A screen shot of a computer screen

AI-generated content may be incorrect.**

**EXPLANATION: -**

**🔹 Variables in the Code:**

**A variable is a name that refers to a value stored in memory. It is used to store data that the program can manipulate.**

**In the given code:**

1. **a**
   * **Assigned the value 5**
   * **This means a stores an integer value.**
2. **b**
   * **Assigned the value "MAYUR"**
   * **This means b stores a string (text) value.**
3. **c**
   * **Assigned the value True**
   * **This means c stores a Boolean (True/False) value.**
4. **d**
   * **Assigned the value 5.6**
   * **This means d stores a floating-point number (decimal) value.**

**🔹 Data Types Identified:**

**A data type tells what kind of value a variable holds. Python has several built-in types. In the output, the types are checked using type().**

1. **a → int (Integer)**
   * **Because it holds a whole number: 5**
2. **b → str (String)**
   * **Because it holds text: "MAYUR"**
3. **c → bool (Boolean)**
   * **Because it holds True, which is one of the two Boolean values (True or False)**
4. **d → float (Floating-point)**
   * **Because it holds a decimal number: 5.6**

**OPERATORS: -**

**INPUT: -**

**A screen shot of a computer

AI-generated content may be incorrect.**

**OUTPUT: -**

**A screen shot of a computer

AI-generated content may be incorrect.**

**EXPLANATION: -**

**🔹 What is an Operator?**

**Operators are special symbols used to perform operations on values or variables. In Python, operators like +, -, \*, /, etc., are used for mathematical calculations.**

**✅ Operators Used in the Code and Their Meaning:**

1. **+ (Addition Operator)**
   * **print(5+5)**
   * **Adds two numbers: 5 + 5 = 10**
2. **- (Subtraction Operator)**
   * **print(2-7)**
   * **Subtracts right number from left: 2 - 7 = -5**
3. **\* (Multiplication Operator)**
   * **print(2\*5)**
   * **Multiplies the numbers: 2 \* 5 = 10**
4. **/ (Division Operator)**
   * **print(14/6)**
   * **Divides and returns a float (decimal): 14 ÷ 6 = 2.333...**
5. **// (Floor Division Operator)**
   * **print(14//6)**
   * **Divides and gives the integer part only (no decimal): 14 ÷ 6 = 2 (ignores remainder)**
6. **\*\* (Exponentiation Operator)**
   * **print(4\*\*2)**
   * **Raises to the power: 4 raised to 2 = 4 × 4 = 16**
7. **% (Modulus Operator)**
   * **print(9%2)**
   * **Gives remainder after division: 9 ÷ 2 has remainder 1**

| **Precedence** | **Operator** | **Meaning** | **Example** | **Result** |
| --- | --- | --- | --- | --- |
| **1 (Highest)** | **\*\*** | **Exponentiation (power)** | **4 \*\* 2** | **16** |
| **2 (MEDIUM)** | **\*, /, //, %** | **Multiplication, Division, Floor Division, Modulus** | **14 / 6, 14 // 6, 9 % 2** | **2.33, 2, 1** |
| **3 (Lowest)** | **+, -** | **Addition, Subtraction** | **5 + 5, 2 - 7** | **10, -5** |

**EXERCISE: - BUILD VERY BASIC CALCULATOR TO PERFORM ADDITION, SUBSTRACTION, MULTIPLICATION, DIVISION**

**INPUT: -**

**A black background with white text

AI-generated content may be incorrect.**

**OUTPUT: -**

**A black screen with white text

AI-generated content may be incorrect.**

**TYPECASTING: -**

**INPUT: -**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**In the given example, a value is stored as a string (text) and another as an integer (number). Since Python does not allow direct addition of a string and an integer, the string must first be converted into an integer before performing the operation. This conversion is known as type casting.**

**The type of type casting used here is called explicit type casting. This means the programmer manually converts one data type to another using a function — in this case, converting the string "5" to the number 5 using the int() function.**

**After conversion, the addition becomes possible, and the result is stored as an integer. At the end, the data types of the original and resulting values are:**

* **The first value is a string (before conversion),**
* **The second value is an integer,**
* **The final result is also an integer.**

**So, this is a clear example of explicit type casting, where a string is intentionally changed to a number to allow a mathematical operation.**

**INPUT: -**

**A screen shot of a computer program

AI-generated content may be incorrect.**

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**In this example, two values are added — one is a whole number (integer), and the other is a decimal number (float). When these two different types are used together in an operation like addition, Python automatically converts the integer to a float to match the other value. This automatic conversion is called implicit type casting.**

**🔹 What is Happening Here?**

* **One value is an integer.**
* **The second value is a float.**
* **When added together, Python does not show an error — instead, it automatically changes the integer into a float.**
* **The result is a float value.**

**🔸 Type of Type Casting:**

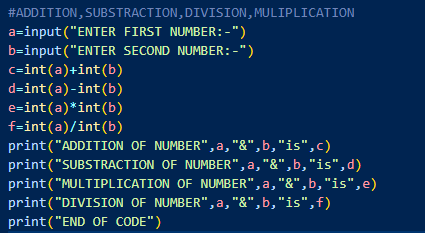
**This is known as implicit type casting because Python handles the conversion on its own, without the programmer writing any function to convert the data types.**

**✅ Summary:**

* **Type casting is used automatically.**
* **It is called implicit type casting.**
* **It happens when combining different data types in an expression (like int + float).**
* **The final result is converted to the more precise type, which is float in this case.**

**TAKE USER INPUT IN PYTHON: -**

**INPUT: -**

****

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**🌟 What is User Input?**

**User input means the program is asking you (the user) to type something using the keyboard while the program is running.  
It helps the program take your own values and then use them in calculations or tasks.  
So instead of using fixed numbers, the program becomes interactive — it works based on what *you* give.**

**🧠 How It Works in This Program (Step by Step):**

**1. Program Asks You for Numbers**

* **When you start the program, it will show a message like "Enter first number".**
* **You type a number (for example: 5) and press Enter.**
* **Then it asks for the second number, and you type another number (for example: 5 again).**

**2. Program Saves What You Typed**

* **Whatever you typed is stored inside the program so it can use it later.**
* **But remember: what you type is first treated as text (even if you type numbers).**

**3. Changing Text to Number**

* **Since the computer can’t do maths with plain text, the program converts the typed text into numbers.**
* **This step is very important — only after this the program can calculate correctly.**

**4. Doing the Math**

* **The program then adds, subtracts, multiplies, and divides the two numbers you gave.**
* **It calculates each answer one by one.**

**5. Showing the Results**

* **After all calculations, the program displays the results:**
  + **What is the sum (addition)?**
  + **What is the difference (subtraction)?**
  + **What is the product (multiplication)?**
  + **What is the result of division?**

**6. Program Ends**

* **Lastly, the program shows a message like “End of code” to tell you everything is done.**

**📌 Why is User Input Important?**

* **Without user input, the program would always work with the same numbers.**
* **With user input, the program works with any numbers you give — so it becomes more useful and flexible.**
* **It also makes the program more fun and interactive, as it responds to your actions.**

**🧾 Example (In Words):**

**Let’s say the program asks:  
➡️ "Enter the first number" → you type 5  
➡️ "Enter the second number" → you type 3**

**The program will now:**

* **Add 5 and 3 → show 8**
* **Subtract 5 and 3 → show 2**
* **Multiply 5 and 3 → show 15**
* **Divide 5 by 3 → show 1.666...**

**✅ In Short:**

* **User input is when the program asks you to type something.**
* **It lets the program take your data and use it.**
* **This makes the program smart, flexible, and more useful.**

**PRINTING OF MULTIPLE LINE STRING: -**

**INPUT: -**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**In Python, when you want to write a string that spans multiple lines, you can use triple quotes. Triple quotes mean using three single quotes (''') or three double quotes (""") at the start and end of the string.**

**This allows you to write the text over several lines exactly as you want it to appear. You can press Enter and continue writing on the next line, and Python will include the line breaks automatically.**

**This method is very useful for writing paragraphs, poems, long messages, or documentation. Whatever you write between the triple quotes will be part of the same string, even if it’s written on different lines.**

**INDEXING IN PYTHON USING STRING: -**

**INPUT: -**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**Explanation of String Indexing in This Code (Easy and Short):**

**In Python, string indexing means accessing characters using their position number in the string.**

**There are two types of indexing:**

**✅ 1. Positive Indexing (left to right):**

* **Starts from 0 for the first character.**
* **Example: In "MAYURESH TELI", index 1 gives "A".**

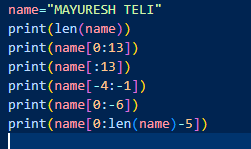
**✅ 2. Negative Indexing (right to left):**

* **Starts from -1 for the last character.**
* **Example: Index -1 gives "I" from the end.**

**Each character in the string "MAYURESH TELI" has both a positive and a negative index.  
Using these index numbers, we can pick any character from the string.**

**SLICING OF STRING: -**

**INPUT: -**

****

**OUTPUT: -**

**A blue screen with white text

AI-generated content may be incorrect.**

**EXPLANATION: -**

**In the given example, the string is "MAYURESH TELI", which has 13 characters.**

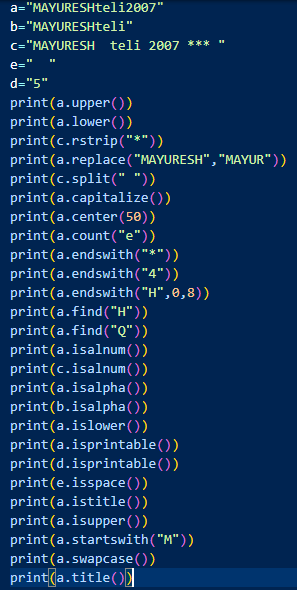
**Here’s what each slice does:**

1. **Length of the string is 13, meaning there are 13 characters in total.**
2. **Slicing from position 0 to 13 gives the full string, because it starts at the beginning and goes till the end (remember, the last number is excluded, so it stops before position 13).**
3. **Slicing from start to 13 (without writing the start index) also gives the full string, because it assumes start is from 0.**
4. **Slicing from 4th last to 1st last (excluding) gives the characters "TEL", which are the characters near the end, but not including the very last one.**
5. **Slicing from beginning to 6th last character gives "MAYURES", meaning it stops before the last 6 characters.**
6. **Slicing from beginning to total length minus 5 gives "MAYURESH", which is everything except the last 5 characters.**

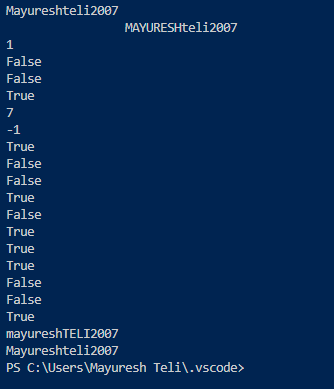
**Each slice uses the position (index) of characters to extract part of the string. This is called slicing, and it helps to work with only the part of the string you need.**

**STRING METHOD IN PYTHON: -**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**🔤 String a = "MAYURESHteli2007"**

1. **a.lower()  
   Converts the entire string to lowercase.  
   → Result is all small letters and digits stay same.**
2. **a.upper()  
   Converts the entire string to uppercase.  
   → All letters become capital, digits stay same.**
3. **a.rstrip("\*")  
   Removes any asterisk \* only from the end of the string.  
   → Since a has no asterisks, it remains unchanged.**

**🧾 String c = "MAYURESH Teli 2007 \*\*\*"**

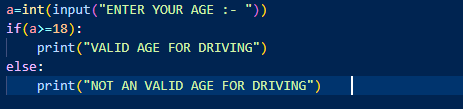
1. **c.replace("MAYURESH", "MAYUR")  
   Replaces the word "MAYURESH" with "MAYUR" only once (first occurrence).  
   → The rest of the string stays the same.**

**🔠 Back to String a**

1. **a.split(" ")  
   Tries to split the string wherever there is a space.  
   → Since there’s no space, the entire string is returned as a single item in a list.**
2. **a.capitalize()  
   Makes only the first letter uppercase and the rest lowercase.  
   → Works even if the string already starts with uppercase.**
3. **a.count("e")  
   Counts how many times the letter e appears in the string.  
   → Only exact lowercase e is counted.**
4. **a.center(50)  
   Centers the string in a space of 50 characters.  
   → It adds equal number of spaces on both sides.**
5. **a.endswith("7")  
   Checks whether the string ends with the digit 7.  
   → Returns true if yes.**
6. **a.startswith("M")  
   Checks whether the string starts with the letter M.  
   → Returns true if yes.**
7. **a.index("t")  
   Finds the first position of the lowercase letter t.  
   → If not found, it gives an error (but here, it exists).**
8. **a.find("h")  
   Also finds the first position of lowercase h.  
   → Same as index(), but returns -1 if not found.**
9. **a.isalnum()  
   Checks if all characters are letters or numbers.  
   → No symbols or spaces should be present.**
10. **a.isalpha()  
    Checks if the entire string contains only letters.  
    → If there are any digits, it gives false.**
11. **a.isascii()  
    Checks if all characters are from the ASCII set (standard keyboard characters).  
    → Returns true if all are English letters, numbers, or symbols.**
12. **a.isdigit()  
    Checks if the entire string contains only numbers.  
    → If there's even one letter, returns false.**
13. **a.islower()  
    Checks if all alphabetic characters are lowercase.  
    → If any capital letter exists, returns false.**
14. **a.isupper()  
    Checks if all alphabetic characters are uppercase.  
    → Returns false if any lowercase letter is present.**
15. **a.isspace()  
    Checks if the string contains only whitespace (spaces, tabs).  
    → Returns false for any visible character.**
16. **a.istitle()  
    Checks if the string follows title case (first letter capital, rest small).  
    → Returns false if not formatted like that.**
17. **a.isprintable()  
    Checks if all characters can be printed (no invisible characters like \n).  
    → Returns true if nothing hidden is in the string.**
18. **b.isprintable()  
    Same check applied to string b = "MAYURESHteli" — and it also returns true.**
19. **a.istitle()  
    Repeating the title case check — returns the same result.**
20. **a.isupper()  
    Repeating the all-uppercase check — same result.**
21. **a.startswith("M")  
    Again checking if the string starts with M — returns true.**
22. **a.swapcase()  
    Reverses the case of all letters.  
    → Capitals become lowercase and lowercase becomes uppercase.**
23. **a.title()  
    Converts the string to title case.  
    → First letter becomes capital, rest lowercase.**

**IF, ELIF & ELSE STATEMENTS: -**

**INPUT: -**

****

**OUTPUT: -**

**A screenshot of a computer screen

AI-generated content may be incorrect.**

**EXPLANATION: -**

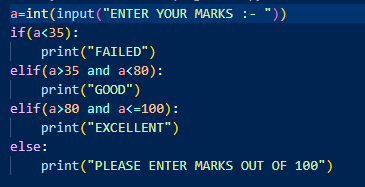
**🔹 Step-by-Step Explanation:**

1. **User Input:**
   * **The program asks the user to enter their age using input("ENTER YOUR AGE :- ").**
   * **int() is used to convert the entered value from text to a number (integer), because age is a number.**
   * **The result is stored in variable a.**
2. **Condition Checking (if-else):**
   * **The if statement checks whether the age is 18 or more using the condition a >= 18.**
     + **If this condition is True, it means the person is old enough to drive.**
     + **It then prints: "VALID AGE FOR DRIVING".**
   * **If the condition is False (meaning age is less than 18), the else block runs.**
     + **It prints: "NOT A VALID AGE FOR DRIVING".**

**🧾 Corrected Output Examples:**

* **If user enters 3:  
  → "NOT A VALID AGE FOR DRIVING" (because 3 is less than 18)**
* **If user enters 45:  
  → "VALID AGE FOR DRIVING" (because 45 is greater than 18)**
* **If user enters 18:  
  → "VALID AGE FOR DRIVING" (because it is equal to 18, so condition is true)**

**INPUT: -**

****

**OUTPUT: -  
A screenshot of a computer program

AI-generated content may be incorrect.**

**EXPLANATION: -**

**Sure! Let's break down the second code example using the same easy and detailed explanation:**

**🔹 What This Program Does:**

**This program takes the marks entered by the user and prints a message based on how many marks they got.**

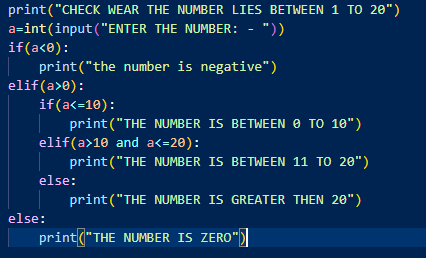
**🧾 Step-by-Step Explanation:**

1. **User Input:**
   * **The program asks: "ENTER YOUR MARKS :- ".**
   * **The input is taken as a string and then converted into an integer using int().**
   * **The value is stored in the variable a.**
2. **Condition Checking with if, elif, and else:**
   * **First condition (if a < 35):**
     + **If the marks are less than 35, it means the user has failed.  
       → Output: "FAILED"**
   * **Second condition (elif a > 35 and a < 80):**
     + **If marks are greater than 35 and less than 80, it prints:  
       → "GOOD"**
   * **Third condition (elif a > 80 and a <= 100):**
     + **If marks are more than 80 and up to 100, it prints:  
       → "EXCELLENT"**
   * **Else (invalid marks):**
     + **If none of the above conditions are true (like if marks are more than 100 or negative), then it prints:  
       → "PLEASE ENTER MARKS OUT OF 100"**

**✅ Correct Output Examples:**

* **If user enters 1:  
  → "FAILED" (because it's less than 35)**
* **If user enters 75:  
  → "GOOD" (between 35 and 80)**
* **If user enters 84:  
  → "EXCELLENT" (between 81 and 100)**
* **If user enters 110:  
  → "PLEASE ENTER MARKS OUT OF 100" (invalid input)**

**INPUT: -**

****

**OUTPUT: -**

**A computer screen shot of a number

AI-generated content may be incorrect.**

**EXPLANATION: -**

**🔹 What the Program Does:**

**This program checks what type of number the user entered:**

* **Whether it's negative, zero, between 0 to 10, between 11 to 20, or greater than 20.**

**🧾 Step-by-Step Explanation:**

1. **User Input:**
   * **It asks: "ENTER THE NUMBER :- ".**
   * **The number entered is taken as a string, converted to integer using int(), and stored in variable a.**
2. **Condition Checking:**
   * **if a < 0**
     + **If the number is less than 0, it is a negative number.  
       → Output: "the number is negative"**
   * **elif a > 0**
     + **If the number is greater than 0, then it goes into a second level of checking:**

**🔹 Nested Conditions:**

* + - **if a <= 10  
      → Output: "THE NUMBER IS BETWEEN 0 TO 10"**
    - **elif a > 10 and a <= 20  
      → Output: "THE NUMBER IS BETWEEN 11 TO 20"**
    - **else  
      → Output: "THE NUMBER IS GREATER THEN 20"**
  + **else (outer one)**
    - **If the number is exactly 0, then this runs:  
      → Output: "THE NUMBER IS ZERO"**

**✅ Output Examples:**

* **If you enter -13:  
  → "the number is negative"**
* **If you enter 11:  
  → "THE NUMBER IS BETWEEN 11 TO 20"**
* **If you enter 45:  
  → "THE NUMBER IS GREATER THEN 20"**
* **If you enter 0:  
  → "THE NUMBER IS ZERO"**

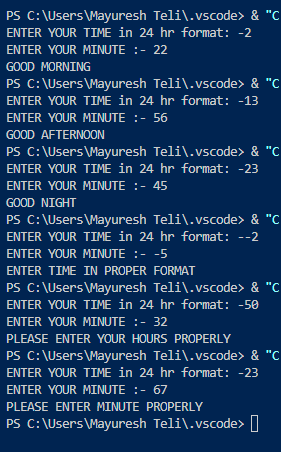
**EXERCISE 2: -- GIVE GREETING TO THE USER ACCORDING TO THE TIME**

**INPUT: -**

**A computer screen with text

AI-generated content may be incorrect.**

**OUTPUT: -**

****

**EXPLANATION: -**

**🔹 What This Program Does:**

**This program takes time input in 24-hour format (hours and minutes) and tells you a greeting based on that time:**

* **Good Morning**
* **Good Afternoon**
* **Good Night**
* **Also checks if the input time is valid or not.**

**🧾 Step-by-Step Explanation:**

1. **User Input:**
   * **It first asks:  
     → "ENTER YOUR TIME IN 24 hr format :- " (stores in variable a)**
   * **Then asks:  
     → "ENTER YOUR MINUTE :- " (stores in variable b)**
2. **Validation Checks:**
   * **if(a<0 or b<0)**
     + **If either hours or minutes are negative, it shows:  
       → "ENTER TIME IN PROPER FORMAT"**
   * **elif(b>60)**
     + **If minutes are more than 60, that’s invalid.  
       → "PLEASE ENTER MINUTE PROPERLY"**
   * **elif(a>24)**
     + **If hours are more than 24, it’s also wrong.  
       → "PLEASE ENTER YOUR HOURS PROPERLY"**

**⏰ Time-based Greetings:**

**After the input is valid, the program checks what time it is and prints an appropriate greeting:**

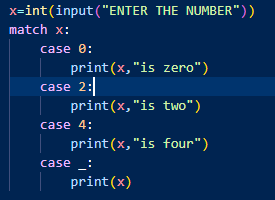
* **elif(a>=0 and a<=12)  
  → "GOOD MORNING"**
* **elif(a>13 and a<=22)  
  → "GOOD AFTERNOON"**
* **else  
  → "GOOD NIGHT"**

**✅ Examples from the Output:**

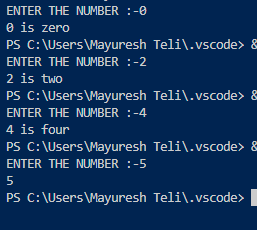
* **If you enter:  
  2 hours and 22 minutes →  
  → "GOOD MORNING"**
* **If you enter:  
  13 hours and 25 minutes →  
  → "GOOD AFTERNOON"**
* **If you enter:  
  23 hours and 25 minutes →  
  → "GOOD NIGHT"**

**MATCH CASE: -**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**📥 INPUT SECTION (Code Explanation):**

**x = int(input("ENTER THE NUMBER"))**

**match x:**

**case 0:**

**print(x, "is zero")**

**case 2:**

**print(x, "is two")**

**case 4:**

**print(x, "is four")**

**case \_:**

**print(x)**

**🔍 Explanation:**

1. **x = int(input(...))**
   * **Takes a number input from the user.**
   * **Converts it from string to integer using int().**
2. **match x:**
   * **This checks the value of x against different cases.**
3. **case 0:, case 2:, case 4:**
   * **These are specific values. If x matches one of them, the corresponding message is printed.**
4. **case \_:**
   * **This is the default case. It runs if none of the above cases match.**
   * **It simply prints the number entered.**

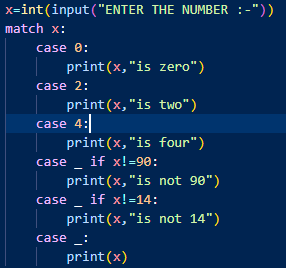
**📤 OUTPUT SECTION (Based on Input Examples):**

* **If user enters 0  
  Output: 0 is zero**
* **If user enters 2  
  Output: 2 is two**
* **If user enters 4  
  Output: 4 is four**
* **If user enters any other number (like -4, -5)  
  Output: It just prints the number itself (e.g., -4, -5), since they don’t match any specific case.**

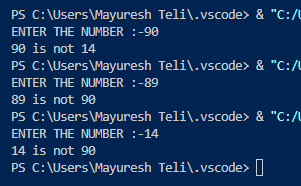
**✅ Summary:**

**This code checks for specific values (0, 2, 4) using match-case and gives a custom message for each. If the value doesn't match any case, it simply prints the number.**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**🔢 INPUT:**

**x = int(input("ENTER THE NUMBER :-"))**

**match x:**

**case 0:**

**print(x, "is zero")**

**case 2:**

**print(x, "is two")**

**case 4:**

**print(x, "is four")**

**case \_ if x != 90:**

**print(x, "is not 90")**

**case \_ if x != 14:**

**print(x, "is not 14")**

**case \_:**

**print(x)**

**🧠 WHAT IT DOES:**

1. **User Input:**
   * **It asks the user to enter a number and stores it as an integer in the variable x.**
2. **match x:**
   * **The program checks the value of x against different cases.**
3. **Cases:**
   * **If x == 0: prints "0 is zero".**
   * **If x == 2: prints "2 is two".**
   * **If x == 4: prints "4 is four".**
   * **If none of the above, it goes to the conditional cases:**
     + **If x != 90: prints "x is not 90".**
     + **If x != 14: prints "x is not 14".**
   * **If none of the conditions match, it just prints x.**

**⚠️ Note: The match-case stops at the first matched case, so only one block is executed.**

**🖨️ OUTPUT EXAMPLES EXPLAINED:**

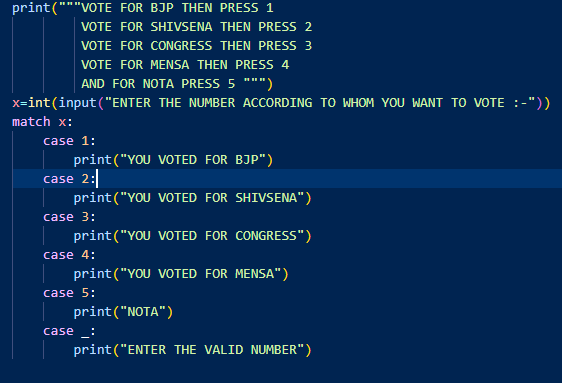
1. **Input: 90**
   * **It doesn't match 0, 2, or 4.**
   * **It skips the if x != 90 block because x == 90.**
   * **Then it matches x != 14, which is True.**
   * **So it prints: 90 is not 14**
2. **Input: -89**
   * **Not 0, 2, 4**
   * **-89 != 90 → True → prints: -89 is not 90**
   * **Match ends here (next blocks skipped)**
3. **Input: -14**
   * **Not 0, 2, 4**
   * **-14 != 90 → True → prints: -14 is not 90**
4. **Input: 14**
   * **Not 0, 2, 4**
   * **14 != 90 → True → prints: 14 is not 90**
   * **But match ends after first true condition, so doesn't check x != 14**

**✅ Summary:**

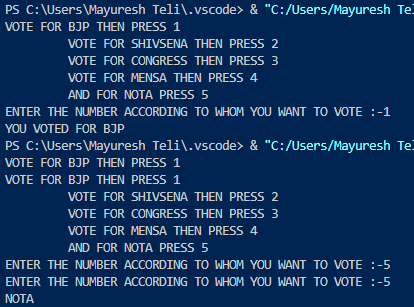
**This code checks for specific values (0, 2, 4), and if none match, it checks whether the input is not equal to 90 or 14 using if conditions inside the match-case structure. Only the first matching case is executed.**

**EXERCISE 3: - MAKE A CODE WHICH COLLCET THE VOTE FROM USER WITHOUT USIN IF ELSE STATEMENT**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**🔢 WHAT THE CODE DOES:**

**This is a voting menu program that lets the user vote for a political party by entering a number.**

**🖥️ CODE WALKTHROUGH:**

**print("""VOTE FOR BJP THEN PRESS 1**

**VOTE FOR SHIVSENA THEN PRESS 2**

**VOTE FOR CONGRESS THEN PRESS 3**

**VOTE FOR MENSA THEN PRESS 4**

**AND FOR NOTA PRESS 5""")**

**🔹 This part shows instructions to the user on what number to press for each political party or NOTA.**

**x = int(input("ENTER THE NUMBER ACCORDING TO WHOM YOU WANT TO VOTE :-"))**

**match x:**

**case 1:**

**print("YOU VOTED FOR BJP")**

**case 2:**

**print("YOU VOTED FOR SHIVSENA")**

**case 3:**

**print("YOU VOTED FOR CONGRESS")**

**case 4:**

**print("YOU VOTED FOR MENSA")**

**case 5:**

**print("NOTA")**

**case \_:**

**print("ENTER THE VALID NUMBER")**

**🔹 This part takes user input and uses the match-case structure to determine and print the selected vote option.**

**🧠 HOW IT WORKS:**

* **If user enters:**
  + **1 → prints "YOU VOTED FOR BJP"**
  + **2 → prints "YOU VOTED FOR SHIVSENA"**
  + **3 → prints "YOU VOTED FOR CONGRESS"**
  + **4 → prints "YOU VOTED FOR MENSA"**
  + **5 → prints "NOTA"**
* **If the user enters anything else (e.g., -5, 7, etc.), it goes to the default case (case \_) and prints "ENTER THE VALID NUMBER"**

**🖨️ OUTPUT EXAMPLES EXPLAINED:**

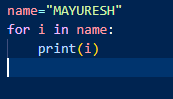
1. **Input: 1  
   → Matches case 1: → prints "YOU VOTED FOR BJP"**
2. **Input: -5  
   → Doesn't match any case → goes to case \_: → prints "ENTER THE VALID NUMBER"**

**✅ SUMMARY:**

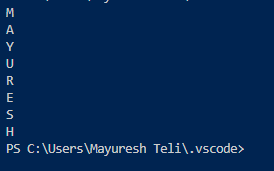
**This code simulates a basic voting machine using match-case. Each number is mapped to a party, and if the input is invalid, it asks the user to enter a valid number.**

**FOR LOOP : -**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**🔤 CODE FUNCTION:**

**name = "MAYURESH"**

**for i in name:**

**print(i)**

**🧠 WHAT IT DOES:**

* **A string "MAYURESH" is stored in the variable name.**
* **A for loop is used to go through each character of the string one by one.**
* **For every character in the string, it prints the character on a new line.**

**🔁 STEP-BY-STEP EXECUTION:**

1. **The loop starts with the first letter 'M' → prints M**
2. **Then 'A' → prints A**
3. **Then 'Y' → prints Y**
4. **Then 'U' → prints U**
5. **Then 'R' → prints R**
6. **Then 'E' → prints E**
7. **Then 'S' → prints S**
8. **Then 'H' → prints H**

**Each letter is printed on a new line because the print() function automatically moves to the next line after printing.**

**📤 OUTPUT:**

**M**

**A**

**Y**

**U**

**R**

**E**

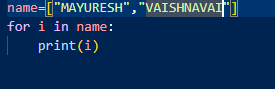
**S**

**H**

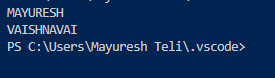
**✅ SUMMARY:**

**This code uses a for loop to print each character of the string "MAYURESH" one by one, each on a separate line. It's a simple way to iterate through a string in Python.**

**INPUT: -**

****

**OUTPUT: -**

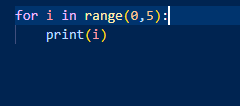
****

**EXPLANATION: -**

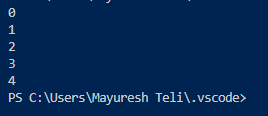
**The program first creates a list called name that has two values: "MAYURESH" and "VAISHNAVAI". Then, using a for loop, it goes through each value in the list one by one. For each value, it prints it on a new line. So first it prints "MAYURESH", and then it prints "VAISHNAVAI".**

**In short, the loop helps to display each name from the list separately.**

**INPUT: -**

****

**OUTPUT: -**

****

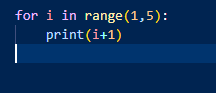
**EXPLANATION: -**

**This program uses a for loop along with the range() function. The range(0, 5) means the loop will start from 0 and go up to 4. It does not include 5.**

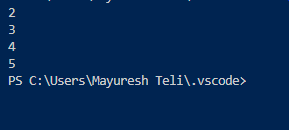
**In each step, the value of i increases by 1, and the current value is printed. So the output will be numbers from 0 to 4 printed one below the other.**

**In short, it prints the numbers 0, 1, 2, 3, and 4 using a loop.**

**INPUT: -**

****

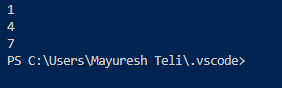
**OUTPUT: -**

****

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**This program uses a for loop with the range() function that has three values: start, stop, and step. It starts from 1, goes up to (but not including) 10, and increases by 3 in each step.**

**So the values will be:**

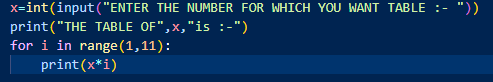
* **Start at 1**
* **Then add 3 → 4**
* **Then add 3 → 7**
* **Next would be 10, but it is not included, so the loop stops at 7**

**The output will be: 1, 4, and 7 — each printed on a new line.**

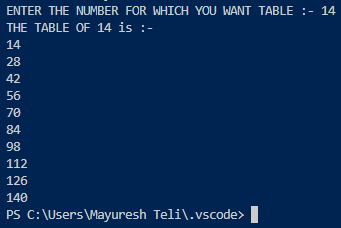
**In short, it prints numbers from 1 to 10 by skipping 2 numbers each time (step of 3).**

**EXERCISE 4: - PRINT TABLE OF ANY NUMBER GIVEN BY THE USER ,BY USING FOR LOOP**

**INPUT: -**

****

**OUTPUT: -**

****

**EXPLANATION: -**

**This program is used to print the multiplication table of any number entered by the user.**

1. **The program first asks the user to enter a number using the input() function.  
   Since the input is in string format, it is converted into an integer using int().**
2. **Then it prints a message saying: “The table of [number] is :-”  
   For example, if the user enters 14, it prints: "THE TABLE OF 14 is :-"**
3. **After that, it uses a for loop with range(1, 11) which means it runs from 1 to 10.**
4. **In every loop, it multiplies the entered number (x) by the loop variable (i) and prints the result.**

**Example:**

**If the user enters 14, the loop will print:**

**14 × 1 = 14  
14 × 2 = 28  
14 × 3 = 42  
...  
up to  
14 × 10 = 140**

**So, the program simply prints the table of the number given by the user from 1 to 10.**