Muppai rojullo Python!🔥

By MASS CODERS - DODAGATTA NIHAR

Tap on the arrow mark below for notes and questions.

▼ Day 1

Why Python?

Python is a popular programming language known for its simplicity, readability, and versatility. Its numerous benefits and features have gained widespread adoption in various fields. Some of the main reasons why Python is widely used are:

1. Web Development
2. Game Development
3. Artificial Intelligence and Machine Learning

Ilantivi dandiga uses unnay Python nerchkunte, so ee language nerchkunte settu!

Where should I run Python Code?

VS Code is Preferable.
Python ni VS Code lo ela run cheyalo telsukovadam kosam ee video chudu:

IG | @niihaaarrrr | Dodagatta Nihar

https://youtu.be/zk5qOQBvuK4

Installation lo emaina difficulties face chestunte, **How to install python for VS code** ani YouTube la search chesi oka sari try chey.
Nen ithe online compilers vaadta ee series mothaniki easy untadi kaabatti beginners kosam.
Online Compiler Link Here: https://www.programiz.com/python-programming/online-compiler/
Meeru chrome lakelli Python Online Compiler ani search chesna sarpoddi, meeku nachina online compiler vaadukondi.

Print Statement in Python

```
print("Nenu Python Muppai rojullo nerchkunta")
```

Output:
Nenu Python Muppai rojullo nerchkunta

Note:
Syntax sakkaga follow avvali appdudaithe ne correct output ostadi.

Possible Mistakes:
Ilanti tappulu asal cheyyak mowa.

- Spelling Mistakes:
  prnt(”Hello World!”)
- Uppercase ‘P’:
  Print(”Hello World!”)
- Missing quotes:
  print(**Hello World!**)

Calculations

Nuv Python ni vaadi addition, subtraction, multiplication, division lu kuda cheyachu!

IG | @niihaaarrrr | Dodagatta Nihar

```
print(2+5)
#Output: 7
print(2-5)
#Output: -3
print(2*5)
#Output: 10
print(6/3)
#Output: 2.0
```

Assigning Variables

```
a = "Nihar"
b = "Loves"
c = "Python"
print(a+" "+b+" "+c)
#Output: Nihar Loves Python
```

```
a = 2
b = 5
print(a+b)
#Output: 7
```

Taking Input from User

Kinda unna code lo user daggarnundi input teeskoni, “a” ane variable lo store chesthanna

```
a = input("Enter your name: ")
print("Your name is "+a)
```

Output:
Enter your name: Dodagatta Nihar
Your name is Dodagatta Nihar

Inthe mowa chaala simple ippudu questions ithe practice chesey settu! Emanna doubt lu osthe telegram community lo adgu nenu, nee mithrulu ready ga untam neeku help cheyyadaniki.

Join Community Here:
https://telegram.oia.bio/masscoders

IG | @niihaaarrrr | Dodagatta Nihar

Questions to Practice (Day 1):

1. Print your name.
2. Print the result of adding two numbers.
3. Print the result of subtracting two numbers.
4. Print the result of multiplying two numbers.
5. Print the result of dividing two numbers.

Congrats! Day 1 successful ga complete chesnav :) Repu kaluddam.

▼ Day 2

Variables

IG | @niihaaarrrr | Dodagatta Nihar

Variables ante containers lekka mawa, avi values ni store cheskuntay!
Nuv vaati values ni eppud kaavalsthe appud maarchachu.

```
a = "Nihar"
print(a) #Output: Nihar
a = "Honey"
print(a) #Output: Honey
```

Data Types

Mukhyanga naalugu data types untay mowa

- String (Example ki “Nihar” , “My Name is Nihar” , “Nen python nerchkunta”)
  Simple terms la cheppalante Stream of Characters.
  • Capital Letters (A – Z)
  • Small Letters (a – z)
  • Digits (0 – 9)
  • Special Characters (~ ! @ # $ % ^ . ?,)
  • Space
- Integer (Example ki 1,2,-9,0)
- Float (Any number with Decimal points, Example ki -3.90, 4.5)
- Boolean (True, False)

```
# Integer data type
my_integer = 42
print(my_integer)  # Output: 42

# Float data type
my_float = 3.14
print(my_float)  # Output: 3.14

# String data type
my_string = "Hello, World!"
print(my_string)  # Output: Hello, World!

# Boolean data type
is_true = True
is_false = False
print(is_true)   # Output: True
print(is_false)  # Output: False
```

Questions to Practice (Day - 2):

IG | @niihaaarrrr | Dodagatta Nihar

1. Declare two variables `a` and `b`, assign integer values to them, and print their sum.
    - **Expected Output:** The sum of `a` and `b`.
2. Create a variable `name` and assign your name to it. Print a greeting message using your name.
    - **Expected Output:** Greeting message with your name, e.g., "Hello, John!"
3. Define a variable `pi` and assign the value of π (pi) to it. Print the value of `pi`.
    - **Expected Output:** The value of π (pi), e.g., 3.14159.
4. Define a variable `is_raining` and ask the user to input either "True" or "False" (as a string). Convert the input to a boolean and print its type.
    - **Expected Input:** "True" or "False"
    - **Expected Output:** The data type of the converted boolean.
5. Create a string variable `sentence` containing any sentence of your choice. Ask the user to input a number, convert it to an integer, and print the sentence repeated that number of times.
    - **Expected Input:** A number (e.g., "3")
    - **Expected Output:** The sentence repeated the specified number of times.
6. Given two variables `x` and `y`, perform the following operations and print the results:
    - Addition of `x` and `y`.
    - Subtraction of `y` from `x`.
    - Multiplication of `x` and `y`.
    - Division of `x` by `y`.
    - `x` raised to the power of `y`.
    - The remainder when `x` is divided by `y`.
    - The floor division of `x` by `y`.

IG | @niihaaarrrr | Dodagatta Nihar

7. Define a variable `value` and assign any numerical value to it. Ask the user to input a new value. Update the variable `value` with the new input and print the updated value.

- **Expected Input:** A numerical value (e.g., "42")
- **Expected Output:** The updated value of the variable.

These questions cover a range of scenarios related to variables and data types in Python. You can use input() function to get user input for interactive questions.

Congrats! Day 2 successful ga complete chesnav :) Repu kaluddam.

▼ Day 3

Order of Instructions

Python line by line code ni execute chestundi mowa

```
print(a)
a = "Naa peru Nihar!"
```

Output:

ERROR!
Traceback (most recent call last):
File "<string>", line 1, in <module>
NameError: name 'a' is not defined

Follow proper syntax

For example, nuv code beginning appudu space asalki ivvakudadu, ala pedte error ochestundi elano chuddam

```
a = 10 + 5
b = 5
 print(a/b)
```

Output:
ERROR!
File "<string>", line 3

IG | @niihaaarrrr | Dodagatta Nihar

print(a/b)
IndentationError: unexpected indent

BODMAS

The BODMAS rule stands for:

1. B - Brackets first
2. O - Orders (exponents and roots, like square roots) next
3. DM - Division and Multiplication, from left to right
4. AS - Addition and Subtraction, from left to right

```
# Example expression: 10 + 5 * (2 ** 3) - 6 / 2

result = 10 + 5 * (2 ** 3) - 6 / 2

# Step 1: Evaluate the expression within the brackets first
#   2 ** 3 = 8
#   So, the expression becomes: 10 + 5 * 8 - 6 / 2

# Step 2: Perform Multiplication
#   5 * 8 = 40
#   So, the expression becomes: 10 + 40 - 6 / 2

# Step 3: Perform division
#   6 / 2 = 3
#   So, the expression becomes: 10 + 40 - 3

# Step 4: Perform addition
#   10 + 40 = 50
#   So, the final result is: 50 - 3 = 47

print(result)  # Output: 47.0
```

```
print(10 / 2 + 3) # Output: 8.0
print(10 / (2 + 3)) # Output: 2.0
```

Questions to practice (Day 3):

What are the expected output of the following expressions?

```
# 1
result = 10 + 3 * 2 - 8 / 4
```

IG | @niihaaarrrr | Dodagatta Nihar

```
# 2
result = 4 ** 2 + 5 / 2 * 3
```

```
# 3
result = (8 + 4) * 3 / 2
```

```
# 4
result = 16 / 4 + 2 ** 3 - 6
```

```
# 5
result = 10 - 3 * (4 + 2) / 5
```

Congrats! Day 3 successful ga complete chesnav :) Repu kaluddam.

▼ Day 4

String Concatenation

Strings ni kalapadaanne string concatenation antaru mowa.

```
a = "Puttaparthi" + " " + "Arrow"
print(a)
# Output: Puttaparthi Arrow
```

Possible Mistakes

Idi only strings ki maatrame vaadtam. Nuv oka string ni oka number ni kalpalevu, ala chesthe error ostundi

```
a = "Hi" + 10
print(a)
```

Output:

ERROR!
Traceback (most recent call last):

IG | @niihaaarrrr | Dodagatta Nihar

File "<string>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str

String Repetition

***** operator ni vaadi strings ni repeat cheyinchachu neek kaavalsinanni saarlu

```
a = "Nihar " * 5
print(a)
```

Output:

Nihar Nihar Nihar Nihar Nihar

```
s = "Arrow"
s = ("* " * 3) + s + (" *" * 3)
print(s)
```

Output:
*** * * Arrow * * ***

Length of String

Nuv string length kanukkovalante **len()** built-in function vaadu mowa.

```
name = input()
length = len(name)
print(length)
```

Output:
Arrow
5

Questions to Solve (Day - 4):

1. Concatenate two strings `str1` and `str2`, and print the result.
   - **Expected Input:** str1 = "Hello", str2 = "World"
   - **Expected Output:** "HelloWorld"
2. Ask the user to enter their name and a greeting. Concatenate the name and greeting to form a personalized message and print it.

IG | @niihaaarrrr | Dodagatta Nihar

- **Expected Input:** name = "John", greeting = "Hi"
- **Expected Output:** "Hi John"

3. Create a string `word` and repeat it 5 times. Print the result.
   - **Expected Input:** word = "Python"
   - **Expected Output:** "PythonPythonPythonPythonPython"
4. Ask the user to enter a word and a number. Repeat the word as many times as the given number and print the result.
   - **Expected Input:** word = "Hello", number = 3
   - **Expected Output:** "HelloHelloHello"
5. Create a string `sentence` and find its length. Print the length of the sentence.
   - **Expected Input:** sentence = "This is a sample sentence."
   - **Expected Output:** 27
6. Ask the user to input a sentence. Find the length of the sentence, and print the last character of the sentence.
   - **Expected Output:** Length of the sentence and the last character.
7. Create two strings `str1` and `str2`. Find the lengths of both strings and concatenate them. Print the concatenated string.
   - **Expected Input:** str1 = "Hello", str2 = "World"
   - **Expected Output:** "HelloWorld"
8. Ask the user to enter two words, `word1` and `word2`. Concatenate the two words with a space in between and print the result.
   - **Expected Input:** word1 = "Hello", word2 = "Python"
   - **Expected Output:** "Hello Python"
9. Create a string `pattern` containing "*" character and repeat it to form a pattern. The pattern should have 5 rows. Print the resulting pattern.
   - **Expected Output:**

```
*
**
***
****
*****
```

IG | @niihaaarrrr | Dodagatta Nihar

Remember to use appropriate string concatenation methods, repeat the strings as required, and utilize the `len()` function to find the length of strings. String manipulation in Python can be fun and useful in various applications!

Congrats! Day 4 successful ga complete chesnav :) Repu kaluddam.

▼ Day 5

Type Conversion

Type conversion python la oka data type nunchi inko data type laki convert cheyyadaniki vaadtam mowa! Idi manam python built-in functions vaadi chestham.
Example ki int(), float(), str() etc.

For example, "123" ane string ni integer laki maarchali ante int() vaadtam

```
num_string = "123"
print(type(num_string))
num_int = int(num_string)
print(type(num_int))
```

Output:

<class 'str'>
<class 'int'>

Paina unna example la int() function **num_string** ni teeskoni, integer loki maarchi **num_int** lo store chestondi.

Alane oka float value ni integer loki maarusthe?
Kinda chudandi 3.14 ni 3 ga ela convert chesamo int() vaadi.

```
num_float = 3.14
num_int = int(num_float)
print(num_int)
print(type(num_int))
```

Output:

IG | @niihaaarrrr | Dodagatta Nihar

3
<class 'int'>

Manam integer and float values in strings ga kuda convert cheyachu, kinda oka sari chuseyyandi.
123 and 3.14 ni string type loki ela convert chestunnamo.

```
num_int = 123
print(type(num_int)) # Output: <class 'int'>
num_string = str(num_int)
print(num_string) # Output: 123
print(type(num_string)) # Output: <class 'str'>


num_float = 3.14
print(type(num_float)) # Output: <class 'float'>
num_string = str(num_float)
print(num_string) # Output: 3.14
print(type(num_string)) # Output: <class 'str'>
```

Questions to practice (Day - 5)

Question 1:
Convert the integer 42 to a string.

Expected Input:

value = 42

Expected Output:

result = "42"

Question 2:
Convert the string "123" to an integer.

Expected Input:

value = "123"

Expected Output:

result = 123

IG | @niihaaarrrr | Dodagatta Nihar

Question 3:
Convert the float 3.14 to an integer.

Expected Input:

value = 3.14

Expected Output:

result = 3

Question 4:
Convert the string "5.5" to a floating-point number.

Expected Input:

value = "5.5"

Expected Output:

result = 5.5

Question 5:
Convert the integer 100 to a boolean.

Expected Input:

value = 100

Expected Output:

result = True

Question 6:
Convert the boolean True to an integer.

Expected Input:

value = True

Expected Output:

result = 1

Question 7:
Convert the string `"False"` to a boolean.

Expected Input:

IG | @niihaaarrrr | Dodagatta Nihar

value = “False”

Expected Output:

result = True

Congrats! Day 5 successful ga complete chesnav :) Repu kaluddam.

▼ Day 6

String Indexing

String Indexing ni manam String lo individual characters ni extract cheyyadaniki vaadtam.
Index of first character - **0**
Index of last character - (**length - 1**)

Example chusey mama neat ga artham avtadi

```
# Define a string
my_string = "Hello, world!"

# Access the first character of the string
first_character = my_string[0]

# Print the result
print(first_character)
```

The output will be `H`.

Negative index vaadi kuda manam characters ni extract cheyyachu. Kinda oka example undi chudandi

```
# Define a string
my_string = "Hello, world!"

# Access the last character of the string
last_character = my_string[-1]

# Print the result
print(last_character)
```

IG | @niihaaarrrr | Dodagatta Nihar

The output will be `!`.

Inkoka example chusey!

```
s = "NIHAR"

# Accessing individual characters using negative indexing
last_char = s[-1]  # 'R'
second_last_char = s[-2]  # 'A'
third_last_char = s[-3]  # 'H'
fourth_last_char = s[-4]  # 'I'
first_char = s[-5]  # 'N'

print("Last character:", last_char)
print("Second last character:", second_last_char)
print("Third last character:", third_last_char)
print("Fourth last character:", fourth_last_char)
print("First character:", first_char)
```

Output:

Last character: R
Second last character: A
Third last character: H
Fourth last character: I
First character: N

String Slicing

Manam string lo oka particular portion ni extract cheyyali ante string slicing vaadtam using `:`

Example chusey mowa!

```
# Define a string
my_string = "Hello, world!"

# Extract the first five characters of the string
substring = my_string[0:5]

# Print the result
print(substring)
```

The output will be `Hello`.

IG | @niihaaarrrr | Dodagatta Nihar

Inkoka example chusey!

```
# Define a string
my_string = "Hello, world!"

# Extract a range of characters from the string
substring = my_string[1:8]

# Print the result
print(substring)
```

The output will be `ello, w`.

Questions to practice (Day 6):

1. Example: Get the first character of the sentence.
   Input: "The sun is shining."
   Output: "T"
2. Example: Get the last character of the sentence.
   Input: "She sells seashells by the seashore."
   Output: "."
3. Example: Get the character at index 3.
   Input: "I love Python!"
   Output: "o"
4. Example: Get the second last character of the sentence.
   Input: "Life is beautiful."
   Output: "l"
5. Example: Get a substring from index 7 to index 14 (exclusive).
   Input: "Welcome to Python programming."
   Output: " to Pyt"
6. Example: Get a substring from index -9 to -3.
   Input: "The future is bright."
   Output: "s brig"
7. Example: Get the first six characters of the sentence.
   Input: "Good things take time."
   Output: "Good t"

IG | @nilihaaarrrr | Dodagatta Nihar

8. Example: Reverse the sentence using slicing.
   Input: "Python is awesome!"
   Output: "!emosewa si nohtyP"
9. Example: Get the length of the sentence using indexing.
   Input: "Coding is fun!"
   Output: 14

Congrats! Day 6 successful ga complete chesnav :) Repu kaluddam.

▼ Day 7

Relational Operators

Veetini manam python lo two values ni compare cheyyadaniki vaadtam.

Ivi motham 6 untay mowa:

- Equal to (==)
- Not equal to (!=)
- Greater than (>)
- Less than (<)
- Greater than or equal to (>=)
- Less than or equal to (<=)

Example chuseyyandi ippudu!

```
x = 5
y = 7

print(x == y)   # False
print(x != y)   # True
print(x > y)    # False
print(x < y)    # True
print(x >= y)   # False
print(x <= y)   # True
```

Note: Double equals sign (==) ni comparison kosam vaadtam, single equals sign (=) ni variable assignment kosam vaadtam.

IG | @niihaaarrrr | Dodagatta Nihar

Idi okasari chudu mowa, output em ostado guess chey

```
a = 5
b = 2-7

print(a == b)
print(a != b)
print(a > b)
print(a < b)
print(a >= b)
print(a <= b)
```

Questions to practice (Day 7):

1. Example: Greater than operator (>)
   Input: 5 > 3
   Output: True
2. Example: Less than operator (<)
   Input: 10 < 20
   Output: True
3. Example: Greater than or equal to operator (>=)
   Input: 7 >= 7
   Output: True
4. Example: Less than or equal to operator (<=)
   Input: 15 <= 12
   Output: False
5. Example: Equal to operator (==)
   Input: "hello" == "hello"
   Output: True
6. Example: Not equal to operator (!=)
   Input: 10 != 20
   Output: True
7. Example: Comparing integers and floats
   Input: 5.0 > 4
   Output: True

IG | @niihaaarrrr | Dodagatta Nihar

8. Example: Comparing strings with different cases
   Input: "Hello" == "hello"
   Output: False
9. Example: Using relational operators with booleans
   Input: True == False
   Output: False
10. Example: Comparing strings
    Input: "apple" < "banana"
    Output: True
11. Example: Comparing None with a string
    Input: None == "Python"
    Output: False
12. Example: Mixing different types in comparisons
    Input: 5 > "3"
    Output: TypeError
13. Example: Using relational operators with negative numbers
    Input: -10 < -5
    Output: True
14. Example: Comparing a string with a number
    Input: "42" == 42
    Output: False
15. Example: Using relational operators with floating-point precision
    Input: 0.1 + 0.1 + 0.1 == 0.3
    Output: False

Congrats! Day 7 successful ga complete chesnav :) Repu kaluddam.

▼ Day 8

Logical Operators

Mukhyanga moodu logical operators untay mowa python lo.
x and y ni operands ankunte

1. `and`: Returns True if both operands are True.
2. `or`: Returns True if at least one of the operands is True.

IG | @niihaaarrrr | Dodagatta Nihar

3. `not`: Returns the opposite boolean value of the operand.

Tip:
True and True ⇒ True
True and False ⇒ False
False and True ⇒ False
False and False ⇒ False

True or True ⇒ True
True or False ⇒ True
False or True ⇒ True
False or False ⇒ False

Examples chud mowa ippudu!

`and` Operator:
The `and` operator returns True if both operands are True. Otherwise, it returns False.

```
# Example 1: Using 'and' with boolean variables
a = True
b = False

result = a and b
print(result)  # Output: False

# Example 2: Using 'and' with expressions
x = 10
y = 5

result = (x > 0) and (y < 10)
print(result)  # Output: True

result = (x > 0) and (y > 10)
print(result)  # Output: False
```

`or` Operator:
The `or` operator returns True if at least one of the operands is True. If both operands are False, it returns False.

```
# Example 1: Using 'or' with boolean variables
a = True
b = False
```

IG | @niihaaarrrr | Dodagatta Nihar

```
result = a or b
print(result)  # Output: True

# Example 2: Using 'or' with expressions
x = 10
y = 5

result = (x > 0) or (y < 10)
print(result)  # Output: True

result = (x < 0) or (y > 10)
print(result)  # Output: False
```

`not` Operator:

The `not` operator returns the opposite boolean value of the operand. If the operand is True, it returns False, and vice versa.

```
# Example 1: Using 'not' with boolean variable
a = True

result = not a
print(result)  # Output: False

# Example 2: Using 'not' with an expression
x = 10
y = 5

result = not (x > y)
print(result)  # Output: False

result = not (x < y)
print(result)  # Output: True
```

Parentheses vaadi more complex logical operations kuda perform cheyyachu okasari look esey kinda.

```
# Complex example using all logical operators
x = 5
y = 10
z = 15

result = (x < y) and (y < z) or (x == z)
print(result)  # Output: True
```

In this example, `(x < y) and (y < z)` evaluates to `True`, and then the whole expression becomes `True or (x == z)`, which is also `True`. Hence, the final output is `True`.

IG | @nilhaaarrrr | Dodagatta Nihar

Questions to practice (Day 8):

1. Example: "and" operator with two True conditions
   Input: (10 > 5) and ("apple" == "apple")
   Output: True
2. Example: "and" operator with one False condition
   Input: (3 < 2) and ("banana" == "orange")
   Output: False
3. Example: "and" operator with one True and one False condition
   Input: (5 >= 3) and (10 != 10)
   Output: False
4. Example: "or" operator with two True conditions
   Input: ("car" == "car") or (7 < 9)
   Output: True
5. Example: "or" operator with one False condition
   Input: ("dog" == "cat") or (6 < 10)
   Output: True
6. Example: "or" operator with both False conditions
   Input: (2 == 3) or (8 > 15)
   Output: False
7. Example: "not" operator with True condition
   Input: not (4 <= 3)
   Output: True
8. Example: "not" operator with False condition
   Input: not ("orange" == "orange")
   Output: False
9. Example: "not" operator with "and" and "or"
   Input: not ((5 > 3) and ("apple" != "banana"))
   Output: False
10. Example: "and" and "not" operators combined
    Input: (10 > 5) and not (3 < 2)
    Output: True

IG | @nihaaarrrr | Dodagatta Nihar

11. Example: "or" and "not" operators combined
    Input: ("cat" == "cat") or not (6 > 10)
    Output: True
12. Example: Using parentheses for grouping expressions
    Input: ((5 >= 3) and (10 != 10)) or (8 > 15)
    Output: False
13. Example: Combining multiple "and" operators
    Input: (2 < 5) and (10 == 10) and ("hello" != "world")
    Output: True
14. Example: Combining multiple "or" operators
    Input: (7 < 3) or (5 >= 5) or ("apple" == "apple")
    Output: True
15. Example: Using "not" operator with an expression
    Input: not (10 > 5 and "car" != "car")
    Output: True
16. Example: Using "not" operator with "or" and "and"
    Input: not (5 > 3 or "dog" == "cat" and 7 < 5)
    Output: False
17. Example: Combining "and" and "or" operators
    Input: (5 > 3 and "apple" != "banana") or (8 == 8 or 6 < 10)
    Output: True
18. Example: Combining "or" and "not" operators
    Input: ("apple" == "banana" or not (6 > 10))
    Output: True
19. Example: Complex combination of "and", "or", and "not"
    Input: not (2 < 5 and (7 > 3 or "hello" == "world"))
    Output: False
20. Example: Nested use of "and", "or", and "not" operators
    Input: (not (5 > 3) and (10 != 10 or "car" == "car"))
    Output: False

Congrats! Day 8 successful ga complete chesnav :)
Repu kaluddam.

▼ Day 9

IG | @niihaaarrrr | Dodagatta Nihar

Conditional Statements

Manam konni conditions use chesi python program flow ni control chestham mowa.
Mukhyanga `if`, `elif` (short for "else if"), and `else`.
Ee statements manaku condition True or False daani batti pain chesthay!

Syntax itluntadi conditional statements di

```
if condition1:
    # Code block executed if condition1 is True
elif condition2:
    # Code block executed if condition1 is False and condition2 is True
else:
    # Code block executed if both condition1 and condition2 are False
```

Detailed explanation chuskunte

Using `if` statement:
The `if` statement is used to execute a block of code if a specified condition is True.

```
# Example 1: Using 'if' to check a condition
age = 18

if age >= 18:
    print("You are an adult.")
```

Output:

```
You are an adult.
```

Using `if` and `else` statements:
The `else` statement is used to execute a block of code when the condition specified in the `if` statement is False.

```
# Example 2: Using 'if' and 'else' to check a condition
age = 15

if age >= 18:
    print("You are an adult.")
```

IG | @niihaaarrrr | Dodagatta Nihar

```
else:
    print("You are a minor.")
```

Output:

```
You are a minor.
```

Using `if`, `elif`, and `else` statements:
The `elif` statement is used to check additional conditions after the `if` condition. It is short for "else if".

```
# Example 3: Using 'if', 'elif', and 'else' to check multiple conditions
age = 25

if age < 18:
    print("You are a minor.")
elif age >= 18 and age < 65:
    print("You are an adult.")
else:
    print("You are a senior citizen.")
```

Output:

```
You are an adult.
```

Decision teeskodaaniki chaala baaga upayoga padtay mawa conditional statements!

Questions to practice (Day 9):

Question 1:
Write a program that takes a number as input and prints "Even" if it's an even number, and "Odd" if it's an odd number.

Expected Input:

```
Enter a number: 7
```

Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
Odd
```

Question 2:
Write a program that takes two numbers as input and prints the larger number.

Expected Input:

```
Enter the first number: 15
Enter the second number: 22
```

Expected Output:

```
22
```

Question 3:
Write a program that takes a character as input and prints "Vowel" if it's a vowel (a, e, i, o, u), and "Consonant" otherwise.

Expected Input:

```
Enter a character: a
```

Expected Output:

```
Vowel
```

Question 4:
Write a program that takes a year as input and prints "Leap Year" if it's a leap year, and "Not a Leap Year" otherwise.

Expected Input:

```
Enter a year: 2024
```

Expected Output:

```
Leap Year
```

IG | @niihaaarrrr | Dodagatta Nihar

Question 5:
Write a program that takes a grade as input (A, B, C, D, or F) and prints "Pass" if it's A, B, C, or D, and "Fail" if it's F.

Expected Input:

```
Enter your grade: C
```

Expected Output:

```
Pass
```

Congrats! Day 9 successful ga complete chesnav :) Repu kaluddam.

▼ Day 10

Nested Conditional Statements

Conditions complex ga unte ivi vaadtam bhayya!

Examples chudandi

```
# Example 1: Nested 'if' statement
x = 10

if x > 0:
    print("x is positive.")
    if x % 2 == 0:
        print("x is even.")
    else:
        print("x is odd.")
else:
    print("x is not positive.")
```

Output:

```
x is positive.
x is even.
```

Explanation:
In this example, we have a nested `if` statement inside the outer `if` statement.

IG | @niihaaarrr | Dodagatta Nihar

The outer `if` checks whether `x` is positive or not. If it is positive, it enters the inner `if` statement to determine if `x` is even or odd. If `x` is not positive, it directly executes the `else` block.

Telugu Explanation: Ee example lo if statement lopala inko if statement undi. Bayata unna if statement condition correct ithe lopala unna code motham execute avtadi lekapothe bayata unna else block execute avtadi.

```
# Example 2: Nested 'if-elif-else' statements
score = 85

if score >= 90:
    print("Grade: A")
elif score >= 80:
    print("Grade: B")
    if score >= 85:
        print("Good job!")
elif score >= 70:
    print("Grade: C")
else:
    print("Grade: Below C")
```

Output:

```
Grade: B
Good job!
```

Explanation:
In this example, we have nested `if-elif-else` statements. The outer `if-elif-else` checks the `score` to assign a grade. If the score is greater than or equal to 90, it prints "Grade: A." If the score is between 80 and 89, it enters the inner `if` statement to check if the score is greater than or equal to 85 and prints "Good job!" along with "Grade: B." If the score is between 70 and 79, it prints "Grade: C." If the score is less than 70, it prints "Grade: Below C."

Telugu Explanation:
Ee example lo score = 85, bayata unna if condition score≥90, idi tappu kaabatti next elif statement ki eltadi. Akkada score ≥80 undi, idi correct ee kaabatti “Grade: B” ani print avtadi. next line lo score≥85 undi, idi kuda correct ee kaabatti “Good job!” ani print avtadi.

IG | @nihaaaarrr | Dodagatta Nihar

Note: Nested conditional statements complex situations handle cheesetappudu chaala baaga use avtadi. Mari ekkuvaga deenni vaadakandi, endukante code ni chadavadaaniki, raayadaaniki, maintain cheyyadaniki ibbandi avtundi.

Questions to practice (Day - 10)

Question 1:
Check if a given number `num` is positive, negative, or zero.

Expected Input:

```
num = 8
```

Expected Output:

```
Positive
```

Question 2:
Determine the type of a given number `num`: even or odd, and whether it is positive or negative.

Expected Input:

```
num = -5
```

Expected Output:

```
Negative
Odd
```

Question 3:
Classify a given `age` into four categories: baby, child, teenager, or adult.

Expected Input:

```
age = 16
```

IG | @niihaaarrrr | Dodagatta Nihar

Expected Output:

```
Teenager
```

Question 4:
Assign a grade to a given `percentage`, considering the following grade scale: A (90-100), B (80-89), C (70-79), D (60-69), and F (below 60).

Expected Input:

```
percentage = 78
```

Expected Output:

```
Grade: C
```

Question 5:
Check if a given `year` is a leap year, and if it is, find the number of days in February for that year.

Expected Input:

```
year = 2000
```

Expected Output:

```
Leap Year, February has 29 days
```

Congrats! Day 10 successful ga complete chesnav :) Repu kaluddam.

▼ Day 11

While Loop

In Python, a `while` loop is used to repeatedly execute a block of code as long as a specified condition is True. The loop continues until the condition becomes False.

IG | @niihaaarrrr | Dodagatta Nihar

Ante okavela ee condition nijamaithe, loop lopala unna statements ni execute cheyyu ani artham. Aa condition tappu ayye varaku lopala unna code execute avthane untadi.

Basic syntax of a `while` loop chuskunte:

```
while condition:
    # Code block to be executed while the condition is True
```

Example 1:

```
# Example: Printing numbers from 1 to 5 using a while loop
num = 1

while num <= 5:
    print(num)
    num += 1
```

Output:

```
1
2
3
4
5
```

Explanation:
In this example, we initialize a variable `num` with the value 1. The `while` loop starts executing when the condition `num <= 5` is True. It prints the value of `num` and then increments it by 1 (`num += 1`). The loop continues executing the code block as long as the condition `num <= 5` remains True. When `num` becomes 6, the condition becomes False, and the loop terminates.

It's important to ensure that the condition in the `while` loop eventually becomes False; otherwise, the loop will run indefinitely, leading to an infinite loop.

Telugu Explanation:
Manam num ani variable lo 1 value store chestunnam. while num≤5 inappudu, lopala unna code execute avtadi. Last lo num value ni +1 increase chestunnam, ante ippudu num = 2. mari 2 kuda 5 kante takkuve kaabatti loop inkosari execute

IG | @niihaaarrrr | Dodagatta Nihar

avtadi. Ila aa condition tappu ayye varaku while block lopala unna code execute avthane untadi.

Example 2:

```
# Example: Finding the sum of numbers from 1 to 10 using a while loop
num = 1
sum_of_numbers = 0

while num <= 10:
    sum_of_numbers += num
    num += 1

print("Sum of numbers from 1 to 10:", sum_of_numbers)
```

Output:

```
Sum of numbers from 1 to 10: 55
```

Explanation:
In this example, we use a `while` loop to find the sum of numbers from 1 to 10. We initialize a variable `num` with the value 1 and a variable `sum_of_numbers` with the initial value 0. The `while` loop starts executing when the condition `num <= 10` is True. Inside the loop, we add the current value of `num` to the `sum_of_numbers` and then increment `num` by 1 (`num += 1`). The loop continues executing the code block as long as the condition `num <= 10` remains True. When `num` becomes 11, the condition becomes False, and the loop terminates.

The loop iterates ten times, adding the numbers 1 to 10 to the `sum_of_numbers`, and then stops when the condition becomes False. The final value of `sum_of_numbers` is printed as the sum of numbers from 1 to 10, which is 55.

Telugu Explanation:
Ee example lo num = 1 and sum_of_numbers = 0 ki initialise chestunnam. num≤10 ayyevarku loop execute avtadi. Prathi iteration lo sum_of_numbers ni num value tho add chestunnam. Loop nunchi bayataki ochaka sum_of_numbers value ni print chestunnam.

Note:

IG | @niihaaarrrr | Dodagatta Nihar

`while` loops are useful when you don't know the exact number of iterations needed in advance, and the loop's termination depends on a certain condition being met. Be cautious while using `while` loops to avoid infinite loops and ensure your loop eventually terminates.

Questions to practice (Day - 11)

Question 1:
Write a while loop that prints numbers from 1 to 10.
Expected Output:

```
1
2
3
4
5
6
7
8
9
10
```

Question 2:
Create a while loop that calculates the sum of numbers from 1 to n, where n is the input.
Expected Input: 5
Expected Output: 15 (1 + 2 + 3 + 4 + 5)

Question 3:
Write a while loop that prints even numbers from 2 to 10.
Expected Output:

```
2
4
6
8
10
```

Question 4:
Create a while loop that keeps prompting the user for a number until they enter a negative number.

IG | @niihaaarrrr | Dodagatta Nihar

Expected Input: 5, 10, -2
Expected Output: (No output, just prompt for input)

Question 5:
Write a while loop that counts down from 10 to 1.
Expected Output:

```
10
9
8
7
6
5
4
3
2
1
```

Question 6:
Create a while loop that asks the user to guess a secret number (e.g., 7) and continues until the correct number is guessed.
Expected Output: (Depends on user input)

Question 7:
Write a while loop that calculates the factorial of a given number.
Expected Input: 4
Expected Output: 24 (4! = 4 * 3 * 2 * 1 = 24)

Question 8:
Create a while loop that prints the Fibonacci series up to n.
Expected Input: 10
Expected Output:

```
0
1
1
2
3
5
8
```

Question 9:
Write a while loop that reads numbers from the user until they enter the number 0. Then, it prints the sum of all the entered numbers.

IG | @nihaaarrrr | Dodagatta Nihar

Expected Input: 3, 5, 2, 0
Expected Output: 10 (3 + 5 + 2)

Question 10:
Create a while loop that prints the square of numbers from 1 to 5.
Expected Output:

```
1
4
9
16
25
```

Congrats! Day 11 successful ga complete chesnav :) Repu kaluddam.

▼ Day 12

For Loop

In Python, a `for` loop is used to iterate over a sequence (such as a list, tuple, string, or range) and execute a block of code for each item in the sequence. The loop continues until all items in the sequence have been processed.

Ante ee particular sequence lo unna prathi okka item ki ee loop lopala unna code ni execute cheyyu ani artham.

The basic syntax of a `for` loop in Python is as follows:

```
for item in sequence:
    # Code block to be executed for each item in the sequence
```

Example chuskunte:

```
# Example: Printing numbers from 1 to 5 using a for loop
for num in range(1, 6):
    print(num)
```

Output:

```
1
2
```

IG | @niihaaarrrr | Dodagatta Nihar

```
3
4
5
```

Explanation:
In this example, we use a `for` loop to iterate over a range of numbers from 1 to 5 (excluding 6). The `range` function generates a sequence of numbers starting from the first argument (1) and up to, but not including, the second argument (6). For each value of `num` in the sequence, the loop executes the code block, which prints the value of `num`.

The loop iterates five times, printing the numbers 1 to 5, and then stops when all items in the sequence have been processed.

Telugu Explanation:
Manam ee for loop lo 1 nunchi 5 varku unna numbers ni print chestunnam. range(1,6) ante 1 nundi 5 varku, 6 excluded ikkada. so 1 nundi 5 varku print avtundi.

Questions to practice (Day - 12):

1. Write a for loop that prints all numbers from 1 to 5.

   Input: None
   Expected Output:

   ```
   1
   2
   3
   4
   5
   ```

2. Use a for loop to calculate the sum of numbers from 1 to 10.

   Input: None
   Expected Output:

   ```
   55
   ```

3. Write a for loop to print each character in the string "Hello".

   Input: None
   Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
H
e
l
l
o
```

4. Write a for loop to print the first 10 even numbers.

   Input: None
   Expected Output:

```
0
2
4
6
8
10
12
14
16
18
```

5. Create a for loop that doubles each number in the sequence: 1, 2, 3, 4, 5.

   Input: None
   Expected Output:

```
2
4
6
8
10
```

Congrats! Day 12 successful ga complete chesnav :)
Repu kaluddam.

▼ Day 13

String Methods

Ante String tho aatalu aadkovadam! Randi aadukundam :)

IG | @niihaaarrrr | Dodagatta Nihar

`len()`: Returns the length of the string.

```
string = "Hello, World!"
length = len(string)
print(length)  # Output: 13
```

`lower()`: Converts all characters in the string to lowercase.

```
string = "Hello, World!"
lower_case = string.lower()
print(lower_case)  # Output: hello, world!
```

`upper()`: Converts all characters in the string to uppercase.

```
string = "Hello, World!"
upper_case = string.upper()
print(upper_case)  # Output: HELLO, WORLD!
```

`strip()`: Removes leading and trailing whitespaces from the string.

```
string = "   Hello, World!   "
stripped_string = string.strip()
print(stripped_string)  # Output: "Hello, World!"
```

`replace()`: Replaces occurrences of a substring with another substring.

```
string = "Hello, World!"
new_string = string.replace("Hello", "Hi")
print(new_string)  # Output: "Hi, World!"
```

`split()`: Splits the string into a list of substrings based on a delimiter.

```
string = "apple,orange,banana"
fruits = string.split(",")
print(fruits)  # Output: ['apple', 'orange', 'banana']
```

`startswith()`: Checks if the string starts with a specific prefix.

```
string = "Hello, World!"
result = string.startswith("Hello")
```

IG | @nihaaarrrr | Dodagatta Nihar

```
print(result)  # Output: True
```

`endswith()`: Checks if the string ends with a specific suffix.

```
string = "Hello, World!"
result = string.endswith("World!")
print(result)  # Output: True
```

`count()`: Returns the number of occurrences of a substring in the string.

```
string = "Hello, World!"
count = string.count("l")
print(count)  # Output: 3
```

`find()`: Returns the index of the first occurrence of a substring. If not found, returns -1.

```
string = "Hello, World!"
index = string.find("o")
print(index)  # Output: 4
```

`isdigit()`: Checks if all characters in the string are digits.

```
string = "12345"
result = string.isdigit()
print(result)  # Output: True
```

`isalpha()`: Checks if all characters in the string are alphabetic.

```
string = "Hello"
result = string.isalpha()
print(result)  # Output: True
```

Inka chaala useful methods unnay mowa ivi kaakunda! Avasaranni batti vaadukuntam.

Questions to practice (Day - 13):

IG | @niihaaarrrr | Dodagatta Nihar

1. Write a Python program that takes a user input string and converts it to lowercase using the lower() method.
   Input: "Hello, World!"
   Expected Output: "hello, world!"
2. Given a string "Hello, World!", find its length using the len() method.
   Input: "Hello, World!"
   Expected Output: 13
3. Write a Python function that takes a string as input and converts it to uppercase using the upper() method.
   Input: "Hello, World!"
   Expected Output: "HELLO, WORLD!"
4. Given a string "Hello123", check if it contains only alphabetic characters using the isalpha() method.
   Input: "Hello123"
   Expected Output: False
5. Write a Python program that takes a user input string with leading and trailing whitespaces and removes them using the strip() method.
   Input: " Python Programming "
   Expected Output: "Python Programming"
6. Given a string "I love Python programming", split it into words using the split() method and print each word on a new line.
   Input: "I love Python programming"
   Expected Output:

```
I
love
Python
programming
```

7. Write a Python program that takes a user input string and replaces all occurrences of "Python" with "Java" using the replace() method.
   Input: "I love Python programming, Python is great."
   Expected Output: "I love Java programming, Java is great."
8. Given a string "Hello, World!", check if it starts with the word "Hello" using the startswith() method.

IG | @niihaaarrrr | Dodagatta Nihar

Input: "Hello, World!"
Expected Output: True

9. Write a Python function that takes a string as input and checks if it ends with the word "end" using the endswith() method.
   Input: "Hello, World!"
   Expected Output: False
10. Given a string "banana", count the occurrences of the letter "a" using the count() method.
    Input: "banana"
    Expected Output: 3
11. Write a Python program that takes a user input string and finds the index of the first occurrence of the letter "l" using the find() method.
    Input: "Hello, World!"
    Expected Output: 2
12. Given a string "12345", check if it consists of only digits using the isdigit() method.
    Input: "12345"
    Expected Output: True

Congrats! Day 13 successful ga complete chesnav :) Repu kaluddam.

▼ Day 14

Nested Loops

Oka loop lopala inko loop pedte nested loop mowa.

Examples chuddam ippudu:

Nested Loop to Print a Rectangular Pattern:

```
# Example: Print a 3x3 rectangular pattern using nested loops
for i in range(3):
    for j in range(3):
        print("*", end=" ")
    print()  # Move to the next line after each row
```

Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
* * *
* * *
* * *
```

Nested Loop to Print a Triangle Pattern:

```
# Example: Print a right-angled triangle pattern using nested loops
n = 5

for i in range(1, n + 1):
    for j in range(i):
        print("*", end=" ")
    print()  # Move to the next line after each row
```

Output:

```
*
* *
* * *
* * * *
* * * * *
```

Ivi baa nerchkunte manam complex data structures ni manam baaga handle cheyachu!

Questions to practice (Day - 14):
Note: Nuv koddiga difficulty ithe face cheyachu ee questions solve chesetappudu, I’ve included questions from Easy to Hard as this topic is very important to handle complex data structures. Try your best! Answers raani vaatiki, you can go through our solutions or copy paste question in chatGPT and give prompt as “Write code in python and explain in detail” for detailed explanation.

1. Write a nested loop that prints a square pattern of stars (asterisks).
   Expected Input: None
   Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
*****
*****
*****
*****
*****
```

2. Create a nested loop to print the multiplication table from 1 to 5 (up to 10 times each).
   Expected Input: None
   Expected Output:

```
1  2  3  4  5
2  4  6  8  10
3  6  9  12  15
4  8  12  16  20
5  10  15  20  25
```

3. Write a nested loop to print a right-angled triangle of numbers in ascending order.
   Expected Input: None
   Expected Output:

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

4. Create a nested loop that prints a hollow square pattern of stars (asterisks).
   Expected Input: None
   Expected Output:

```
*****
*   *
*   *
*   *
*****
```

5. Write a nested loop to calculate and print the sum of numbers from 1 to 5 using nested iteration.

IG | @niihaaarrrr | Dodagatta Nihar

Expected Input: None
Expected Output: 15

6. Create a nested loop to print the reverse of the multiplication table from 1 to 5 (up to 10 times each).
   Expected Input: None
   Expected Output:

```
5  4  3  2  1
10 8  6  4  2
15 12 9  6  3
20 16 12 8  4
25 20 15 10 5
```

7. Write a nested loop that prints a right-angled triangle of stars (asterisks).
   Expected Input: None
   Expected Output:

```
*
**
***
****
*****
```

8. Create a nested loop to find and print prime numbers from 2 to 20.
   Expected Input: None
   Expected Output: 2 3 5 7 11 13 17 19
9. Write a nested loop that prints a diamond pattern of stars (asterisks).
   Expected Input: None
   Expected Output:

```
  *
 ***
*****
 ***
  *
```

10. Create a nested loop to find and print the factorial of numbers from 1 to 5.
    Expected Input: None
    Expected Output: 1 2 6 24 120

IG | @niihaaarrrr | Dodagatta Nihar

Congrats! Day 14 successful ga complete chesnav :) Repu kaluddam.

▼ Day 15

Loop Control Statements

Mukhyanga mooduntay mowa ivi: `break`, `continue`, and `pass`.

`break` Statement:
The `break` statement is used to terminate a loop prematurely when a certain condition is met.

```
# Example: Using 'break' to stop the loop when the value reaches 3
for num in range(1, 6):
    if num == 3:
        break
    print(num)
```

Output:

```
1
2
```

Explanation:
In this example, the loop runs from 1 to 5, but when `num` becomes 3, the `break` statement is encountered, and the loop is terminated immediately.
Telugu Explanation:
Ikkada loop 1 to 5 varku execute avvali mowa, kaani okavela num = 3 indi ankondi, break statement undadam valla loop nundi bayataku ochestadi.

`continue` Statement:
The `continue` statement is used to skip the current iteration of a loop and move to the next iteration.

```
# Example: Using 'continue' to skip printing even numbers
for num in range(1, 6):
    if num % 2 == 0:
        continue
    print(num)
```

IG | @niihaaarrrr | Dodagatta Nihar

Output:

```
1
3
5
```

Explanation:
In this example, the loop runs from 1 to 5, but when an even number is encountered (2 and 4), the `continue` statement is used, skipping the `print` statement, and the loop moves to the next iteration.
Telugu Explanation:
Ikkada loop 1 nundi 5 varku execute itadi mowa, kaani madyalo even numbers (2 and 4) indi ankondi, continue statement undadam valla, next unna print statement skip ipoyyi, next iteration loki elpotadi.

`pass` Statement:
The `pass` statement is used as a placeholder when you don't want to execute any code inside a loop or a conditional block.

```
# Example: Using 'pass' to do nothing inside the loop
for num in range(1, 6):
    pass
```

Explanation:
In this example, the loop runs from 1 to 5, but the `pass` statement does nothing inside the loop.
Telugu Explanation:
pass unte loop lopala unna code edi execute avvad mowa simple.

Questions to practice (Day - 15):

1. Write a Python program to print all numbers from 1 to 10, but stop the loop immediately when reaching 5 using the `break` statement.

   Expected Output:

   ```
   1
   2
   3
   4
   ```

IG | @niihaaarrrr | Dodagatta Nihar

2. Given a list of numbers [1, 2, 3, 4, 5], use a `for` loop to print the elements one by one. However, if the element is 3, skip it using the `continue` statement.
   Note: Mowa nuv ee problem lo list ni create cheyyali ante my_list = [1,2,3,4,5] ani raayi, in detailed lists gurinchi further reels lo maatladta.

   Expected Output:

```
1
2
4
5
```

3. Write a Python function that takes a string as input and checks if it contains the letter 'o'. If it does, print "Found 'o'" and use the `break` statement to stop searching.

   Input: "Hello, World!"
   Expected Output:

```
Found 'o'
```

4. Given a list of numbers [1, 2, 3, 4, 5], use a `for` loop to double each element and print the result. However, if the element is 4, use the `continue` statement to skip it.

   Expected Output:

```
2
4
6
10
```

5. Write a Python program to print all numbers from 1 to 20 using a `while` loop. However, stop the loop when reaching 15 using the `break` statement.

   Expected Output:

```
1
2
3
... (up to 15)
```

IG | @niihaaarrrr | Dodagatta Nihar

Congrats! Day 15 successful ga complete chesnav :) Repu kaluddam.

▼ Day 16

Comparing Strings in Python

Ante Strings ni compare cheyyadam mowa!

Equality and Inequality Comparison (`==`, `!=`):

```
# Example: Equality and inequality comparison
string1 = "hello"
string2 = "Hello"

if string1 == string2:
    print("Both strings are equal.")
else:
    print("The strings are not equal.")

if string1 != string2:
    print("The strings are not equal.")
else:
    print("Both strings are equal.")
```

Output:

```
The strings are not equal.
The strings are not equal.
```

Lexicographical order ante dictionaries lo ela order lo words untayo atla!

Example 2:

```
# Example: Lexicographical comparison
string1 = "apple"
string2 = "banana"

if string1 < string2:
    print("string1 comes before string2.")
else:
    print("string1 comes after or is equal to string2.")

if string1 <= string2:
    print("string1 comes before or is equal to string2.")
else:
```

IG | @niihaaarrrr | Dodagatta Nihar

```
    print("string1 comes after string2.")

if string1 > string2:
    print("string1 comes after string2.")
else:
    print("string1 comes before or is equal to string2.")

if string1 >= string2:
    print("string1 comes after or is equal to string2.")
else:
    print("string1 comes before string2.")
```

Output:

```
string1 comes before string2.
string1 comes before or is equal to string2.
string1 comes before or is equal to string2.
string1 comes before string2.
```

Questions to practice (Day - 16):

1. Given two strings "Python" and "python", check if they are equal using the equality operator (`==`).

   Input: "Python", "python"
   Expected Output: False

2. Write a Python function that takes a string as input and checks if it is an empty string using the equality operator (`==`).

   Input: ""
   Expected Output: True

Congrats! Day 16 successful ga complete chesnav :) Repu kaluddam.

▼ Day 17

Lists

Oka collection of items ni store cheskovadaniki list ni ithe vaadtam mowa!
Lists are mutable, ante nuv daanni modify cheskovachu ante kotha elements ni add cheyachu, avasaram lenivi teeseyachu, alaaaaa!
List lo unna items ni square brackets `[ ]` lo esi pedtam, separated by a comma.

IG | @niihaaarrrr | Dodagatta Nihar

Examples chuskunte:

Creating Lists:
You can create a list by enclosing elements in square brackets.

```
# Example: Creating a list of numbers
numbers = [1, 2, 3, 4, 5]

# Creating a list of strings
fruits = ["apple", "banana", "orange"]

# Creating a mixed-type list
mixed_list = [1, "apple", True, 3.14]
```

Accessing Elements:

```
# Example: Accessing elements in a list
fruits = ["apple", "banana", "orange"]

print(fruits[0])  # Output: "apple"
print(fruits[2])  # Output: "orange"
```

Modifying Elements:

```
# Example: Modifying elements in a list
fruits = ["apple", "banana", "orange"]
fruits[1] = "grape"
print(fruits)  # Output: ['apple', 'grape', 'orange']
```

List Operations:

```
# Example: List operations
list1 = [1, 2, 3]
list2 = [4, 5, 6]

# Concatenation
result = list1 + list2
print(result)  # Output: [1, 2, 3, 4, 5, 6]

# Repetition
repeated_list = list1 * 3
print(repeated_list)  # Output: [1, 2, 3, 1, 2, 3, 1, 2, 3]
```

List Methods:

IG | @niihaaarrrr | Dodagatta Nihar

```
# Example: List methods
fruits = ["apple", "banana", "orange"]

# Adding elements
fruits.append("grape")
print(fruits)  # Output: ['apple', 'banana', 'orange', 'grape']

# Removing elements
fruits.remove("banana")
print(fruits)  # Output: ['apple', 'orange', 'grape']

# Sorting elements
fruits.sort()
print(fruits)  # Output: ['apple', 'grape', 'orange']
```

Lists are widely used in Python for storing and manipulating data. They are flexible, efficient, and an essential tool for various programming tasks. Understanding lists and their methods will help you work with collections of data effectively in Python.

Questions to practice (Day - 17):

1. Write a Python program that takes a list of numbers [1, 2, 3, 4, 5] and prints each number on a new line.

   Input: [1, 2, 3, 4, 5]
   Expected Output:

   ```
   1
   2
   3
   4
   5
   ```

2. Given a list of strings ["apple", "banana", "orange"], concatenate all the strings together with a space in between and print the result.

   Input: ["apple", "banana", "orange"]
   Expected Output: "apple banana orange"

3. Write a Python function that takes a list of numbers as input and returns the sum of all the numbers.

IG | @niihaaarrrr | Dodagatta Nihar

Input: [1, 2, 3, 4, 5]
Expected Output: 15

4. Given a list of numbers [10, 20, 30, 40, 50], find the maximum number using the `max()` function and print the result.

   Input: [10, 20, 30, 40, 50]
   Expected Output: 50

5. Write a Python program that takes a list of names ["Alice", "Bob", "Charlie"] and checks if a given name (e.g., "Alice") is present in the list. Print "Name found" if the name is in the list; otherwise, print "Name not found".

   Input: Names = ["Alice", "Bob", "Charlie"], Name = "Alice"
   Expected Output: "Name found"

Congrats! Day 17 successful ga complete chesnav :) Repu kaluddam.

▼ Day 18

Tuples

Idi kuda list laaga oka data structure eyy! kaani tuples are immutable ante okasari deenni create chesnaka elements ni nuv add or remove cheyyalev ani. Elements anni parentheses `()` lo esi pedtam seperated by a comma.

Example chusthe:

Creating Tuples:

```
# Example: Creating a tuple
fruits = ("apple", "banana", "orange")
coordinates = (3.14, 2.71)
```

Accessing Elements:

```
# Example: Accessing elements in a tuple
fruits = ("apple", "banana", "orange")

print(fruits[0])  # Output: "apple"
print(fruits[2])  # Output: "orange"
```

IG | @niihaaarrrr | Dodagatta Nihar

Tuple Packing and Unpacking:

```
# Example: Tuple packing and unpacking
person = ("John", 30, "New York")
name, age, city = person

print(name)  # Output: "John"
print(age)   # Output: 30
print(city)  # Output: "New York"
```

Tuple Functions:
Tuples support various built-in functions like `len()`, `min()`, and `max()`.

```
# Example: Tuple functions
numbers = (5, 2, 8, 1, 7)

# Length of the tuple
length = len(numbers)
print(length)  # Output: 5

# Maximum and minimum elements in the tuple
maximum = max(numbers)
minimum = min(numbers)
print(maximum, minimum)  # Output: 8 1
```

Tuple Concatenation and Repetition:

```
# Example: Tuple concatenation and repetition
tuple1 = (1, 2, 3)
tuple2 = (4, 5, 6)

# Concatenation
result = tuple1 + tuple2
print(result)  # Output: (1, 2, 3, 4, 5, 6)

# Repetition
repeated_tuple = tuple1 * 3
print(repeated_tuple)  # Output: (1, 2, 3, 1, 2, 3, 1, 2, 3)
```

Tuples are handy when you need to store data that should remain unchanged throughout your program's execution. Their immutability makes them suitable for certain use cases, providing a safe and efficient way to group related data.

Questions to practice (Day - 18):

IG | @niihaaarrrr | Dodagatta Nihar

1. Create a tuple containing three elements: 'apple', 5, and True.
   Expected Input: No input required.
   Expected Output: ('apple', 5, True)
2. Access the second element from the given tuple: ('cat', 'dog', 'bird', 'fish').
   Expected Input: No input required.
   Expected Output: 'dog'
3. Concatenate two tuples: (1, 2, 3) and ('a', 'b', 'c').
   Expected Input: No input required.
   Expected Output: (1, 2, 3, 'a', 'b', 'c')
4. Find the length of the tuple: (10, 20, 30, 40, 50).
   Expected Input: No input required.
   Expected Output: 5
5. Check if the element 25 exists in the tuple: (10, 20, 30, 40, 50).
   Expected Input: No input required.
   Expected Output: False
6. Create a new tuple with elements from the given tuple (3, 6, 9) repeated 3 times.
   Expected Input: No input required.
   Expected Output: (3, 6, 9, 3, 6, 9, 3, 6, 9)
7. Perform packing and unpacking on a tuple containing the names of three fruits: 'apple', 'banana', and 'orange'. Use unpacking to assign each fruit to three variables: fruit1, fruit2, and fruit3.

   Expected Input: No input required.
   Expected Output:

- fruit1 = 'apple'
- fruit2 = 'banana'
- fruit3 = 'orange'

Congrats! Day 18 successful ga complete chesnav :)
Repu kaluddam.

▼ Day 19

Dictionaries

IG | @nilhaaarrrr | Dodagatta Nihar

Dictionaries key-value pairs ni store cheskuntadi mowa.
Dictionaries are mutable, ante nuv veetini modify kuda cheskovachu.
Dictionaries are defined using curly braces `{ }`, and each key-value pair is separated by a colon `:`.

Examples chuskunte:

Creating Dictionaries:
You can create a dictionary by enclosing key-value pairs in curly braces.

```
# Example: Creating a dictionary of student information
student = {
    "name": "Dodagatta Nihar",
    "age": 25,
    "major": "Computer Science",
    "gpa": 3.8
}
```

Accessing Values:
You can access values in a dictionary using keys. Dictionary keys are unique.

```
# Example: Accessing values in a dictionary
student = {
    "name": "Dodagatta Nihar",
    "age": 25,
    "major": "Computer Science",
    "gpa": 3.8
}

print(student["name"])  # Output: "Dodagatta Nihar"
print(student["gpa"])   # Output: 3.8
```

Questions to practice (Day - 19):

1. Question: Create an empty dictionary.
   Expected Output: {}
2. Question: Create a dictionary to store the age of two people, "John" and "Alice." John is 25 years old, and Alice is 30 years old.
   Expected Output: {"John": 25, "Alice": 30}

IG | @niihaaarrrr | Dodagatta Nihar

3. Question: Access the value associated with the key "city" from the given dictionary.
   Expected Input: Dictionary: {"name": "Alice", "city": "New York", "age": 30}
   Expected Output: "New York"
4. Question: Create a dictionary to store the contact information of a person. The person's name is "Bob," and their email is "bob@example.com."
   Expected Output: {"name": "Bob", "email": "bob@example.com"}
5. Question: Access the value associated with the key "score" from the given dictionary.
   Expected Input: Dictionary: {"name": "John", "age": 22, "score": 85}
   Expected Output: 85
6. Question: Create a dictionary to represent a rectangle. The rectangle has a width of 10 and a height of 5.
   Expected Output: {"width": 10, "height": 5}
7. Question: Access the value associated with the key "phone" from the given dictionary. If the key does not exist, return "Not available."
   Expected Input: Dictionary: {"name": "Eve", "age": 27}
   Expected Output: "Not available"

Congrats! Day 19 successful ga complete chesnav :) Repu kaluddam.

▼ Day 20

Working with Dictionaries

Modifying and Adding Key-Value Pairs:

```
# Example: Modifying and adding key-value pairs
student = {
    "name": "Dodagatta Nihar",
    "age": 25,
    "major": "Computer Science",
    "gpa": 3.8
}

# Modifying value
student["gpa"] = 4.0

# Adding new key-value pair
student["university"] = "XYZ University"

print(student)
```

IG | @niihaaarrrr | Dodagatta Nihar

```
# Output: {'name': 'Dodagatta Nihar', 'age': 25, 'major': 'Computer Science', 'gp
a': 4.0, 'university': 'XYZ University'}
```

Dictionary Methods:
Python provides various built-in methods to perform common operations on dictionaries.

```
# Example: Dictionary methods
student = {
    "name": "Dodagatta Nihar",
    "age": 25,
    "major": "Computer Science",
    "gpa": 3.8
}

# Get keys and values
keys = student.keys()
values = student.values()

print(keys)   # Output: dict_keys(['name', 'age', 'major', 'gpa'])
print(values) # Output: dict_values(['Dodagatta Nihar', 25, 'Computer Science', 3.
8])

# Check if a key exists in the dictionary
if "major" in student:
    print("Major:", student["major"])  # Output: Major: Computer Science

# Remove a key-value pair from the dictionary
removed_value = student.pop("age")
print("Removed Value:", removed_value)  # Output: Removed Value: 25

print(student)  # Output: {'name': 'Dodagatta Nihar', 'major': 'Computer Science',
'gpa': 3.8}
```

Questions to practice (Day - 20):

1. Question: Given the initial dictionary `employee` as `{"name": "Alice", "age": 30}`, modify the value of the key "age" to 35.
   Expected Input: `employee = {"name": "Alice", "age": 30}`
   Expected Output: `{"name": "Alice", "age": 35}`
2. Question: Add a new key-value pair to the dictionary `fruits`, where the key is "orange" and the value is 3.

IG | @niihaaarrrr | Dodagatta Nihar

Expected Input: `fruits = {"apple": 5, "banana": 7}`
Expected Output: `{"apple": 5, "banana": 7, "orange": 3}`

3. Question: Given the dictionary `inventory`, remove the key "sugar" and its associated value from the dictionary.
   Expected Input: `inventory = {"apple": 10, "banana": 15, "sugar": 2}`
   Expected Output: `{"apple": 10, "banana": 15}`
4. Question: Create a function called `add_stock` that takes a dictionary `stock` and an item `item_name` (string) as input and adds 1 to the value of the corresponding key in the dictionary. Return the modified dictionary.
   Expected Input: `stock = {"apple": 10, "banana": 15}`, `item_name = "banana"`
   Expected Output: `{"apple": 10, "banana": 16}`
5. Question: Given the dictionary `scores`, check if the key "Alice" exists. If it exists, print the associated value; otherwise, print "Key not found."
   Expected Input: `scores = {"Bob": 85, "Charlie": 90, "Alice": 78}`
   Expected Output: `78`
6. Question: Write a function called `count_vowels` that takes a string `text` as input and returns a dictionary containing the count of each vowel (a, e, i, o, u) in the text. Ignore case sensitivity.
   Expected Input: `text = "Hello, World!"`
   Expected Output: `{"a": 0, "e": 1, "i": 0, "o": 2, "u": 0}`
7. Question: Given the dictionary `student_grades`, find the highest grade and its corresponding student name. The dictionary contains student names as keys and their grades as values.
   Expected Input: `student_grades = {"Alice": 85, "Bob": 90, "Charlie": 78}`
   Expected Output: `("Bob", 90)`

Congrats! Day 20 successful ga complete chesnav :) Repu kaluddam.

▼ Day 21

Sets

Sets ante oka unordered collection of unique elements mowa, ante oka element set ki rendu rendu moodu moodu undav okate untadi and then ee particular order lone align ayyi undali ee elements ani em ledu, vaatiki istam ochinattu avi undachu.

IG | @hiihaaarrrr | Dodagatta Nihar

In Python, sets are defined using curly braces `{}` or by using the `set()` constructor.

Creating a Set:

```
# Using curly braces
my_set = {1, 2, 3, 4, 5}

# Using the set() constructor
another_set = set([5, 6, 7, 8, 9])
```

Adding Elements to a Set:

```
my_set = {1, 2, 3}
my_set.add(4)
my_set.add(5)
print(my_set)  # Output: {1, 2, 3, 4, 5}
```

Removing Elements from a Set:

```
my_set = {1, 2, 3, 4, 5}
my_set.remove(3)
print(my_set)  # Output: {1, 2, 4, 5}

# Remove an element without raising an error if the element is not present
my_set.discard(10)

# Pop removes and returns an arbitrary element from the set
popped_element = my_set.pop()
print(popped_element, my_set)  # Output (random order): 1, {2, 4, 5}
```

Questions to practice (Day - 21):

Question 1: Creating a set
Expected Input: None
Expected Output: An empty set

Question 2: Adding elements to a set
Expected Input: Set: {1, 2, 3}, Element to add: 4
Expected Output: Set: {1, 2, 3, 4}

IG | @niihaaarrrr | Dodagatta Nihar

Question 3: Removing an element from a set
Expected Input: Set: {1, 2, 3, 4}, Element to remove: 3
Expected Output: Set: {1, 2, 4}

Question 4: Removing a non-existent element from a set
Expected Input: Set: {1, 2, 3, 4}, Element to remove: 5
Expected Output: Set: {1, 2, 3, 4}

Question 5: Creating a set from a list
Expected Input: List: [5, 10, 15, 20]
Expected Output: Set: {5, 10, 15, 20}

Question 6: Adding multiple elements to a set
Expected Input: Set: {1, 2, 3}, Elements to add: {4, 5}
Expected Output: Set: {1, 2, 3, 4, 5}

Question 7: Removing elements using discard()
Expected Input: Set: {10, 20, 30, 40}, Elements to remove: {20, 50}
Expected Output: Set: {10, 30, 40}

Congrats! Day 21 successful ga complete chesnav :) Repu kaluddam.

▼ Day 22

Working with Sets

Set Operations:

```
set1 = {1, 2, 3, 4}
set2 = {3, 4, 5, 6}

# Union of two sets
union_set = set1.union(set2)
print(union_set)  # Output: {1, 2, 3, 4, 5, 6}

# Intersection of two sets
intersection_set = set1.intersection(set2)
print(intersection_set)  # Output: {3, 4}

# Difference between two sets
difference_set = set1.difference(set2)
print(difference_set)  # Output: {1, 2}

# Symmetric Difference (elements in either set, but not in both)
symmetric_difference_set = set1.symmetric_difference(set2)
print(symmetric_difference_set)  # Output: {1, 2, 5, 6}
```

IG | @nihaaarrr | Dodagatta Nihar

Membership Test:

```
my_set = {1, 2, 3, 4, 5}
print(3 in my_set)  # Output: True
print(6 not in my_set)  # Output: True
```

Questions to practice (Day - 22):

Question 1: Symmetric Difference between two sets
Expected Input: Set A: {1, 2, 3}, Set B: {3, 4, 5}
Expected Output: {1, 2, 4, 5}

Question 2: Checking membership in a set
Expected Input: Set: {1, 2, 3}, Element to check: 2
Expected Output: True

Question 3: Union of two sets
Expected Input: Set A: {1, 2, 3}, Set B: {3, 4, 5}
Expected Output: {1, 2, 3, 4, 5}

Question 4: Intersection of two sets
Expected Input: Set A: {1, 2, 3}, Set B: {3, 4, 5}
Expected Output: {3}

Question 5: Difference between two sets
Expected Input: Set A: {1, 2, 3, 4}, Set B: {3, 4, 5}
Expected Output: {1, 2}

Question 6: Removing duplicate elements from a list using a set
Expected Input: List: [1, 2, 2, 3, 4, 4, 5]
Expected Output: [1, 2, 3, 4, 5]

Question 7: Finding the length of a set
Expected Input: Set: {1, 2, 3, 4, 5}
Expected Output: 5

Question 8: Checking if one set is a subset of another
Expected Input: Set A: {1, 2, 3}, Set B: {1, 2, 3, 4, 5}
Expected Output: True

Question 9: Checking if two sets are disjoint
Expected Input: Set A: {1, 2, 3}, Set B: {4, 5, 6}

IG | @nihaaarrr | Bodagatta Nihar

Expected Output: True

Question 10: Removing all elements from a set
Expected Input: Set: {1, 2, 3, 4, 5}
Expected Output: An empty set

Congrats! Day 22 successful ga complete chesnav :) Repu kaluddam.

▼ Day 23

Functions Part - 1

Functions ni vaadi manam reusable code ni ithe create cheyachu mowa.
Functions manaku code ni proper ga organise cheskodaniki upayogapadtay!

Creating a Function:
To define a function, you use the `def` keyword, followed by the function name, a pair of parentheses `()`, and a colon `:`. The code block inside the function is indented and contains the instructions that the function will execute when called.

```
def greet():
    print("Hello, World!")

# Calling the function
greet()  # Output: Hello, World!
```

Function Parameters:
You can define parameters inside the parentheses to pass data into the function. Parameters act as placeholders for the actual values that you provide when calling the function.

```
def greet_user(name):
    print(f"Hello, {name}!")

# Calling the function with an argument
greet_user("Alice")  # Output: Hello, Alice!
```

Return Statement:
Functions can return values using the `return` keyword. When a function returns a value, you can capture it and use it elsewhere in your code.

IG | @nihaaarrr | Dodagatta Nihar

```
def add_numbers(a, b):
    return a + b

# Calling the function and storing the result in a variable
result = add_numbers(3, 5)
print(result)  # Output: 8
```

Questions to practice (Day - 23):

Problem 1: Simple function without parameters
Write a function called `greet` that takes no parameters and returns the string "Hello, World!"

Expected Output:

```
message = greet()
print(message)  # Output: Hello, World!
```

Problem 2: Function with one parameter
Write a function called `greet_user` that takes a `name` parameter and returns a greeting message with the name.

Expected Output:

```
message = greet_user("Alice")
print(message)  # Output: Hello, Alice!
```

Problem 3: Function with two parameters and mathematical operation
Write a function called `add_numbers` that takes two parameters `a` and `b`, and returns their sum.

Expected Output:

```
result = add_numbers(5, 10)
print(result)  # Output: 15
```

Problem 4: Function with a conditional statement
Write a function called `check_even_odd` that takes a `number` parameter and returns

IG | @nihaaarrrr | Dodagatta Nihar

"Even" if the number is even, and "Odd" if the number is odd.

Expected Output:

```
result = check_even_odd(7)
print(result)  # Output: Odd
```

Problem 5: Function with default parameter value

Write a function called `greet_user` that takes an optional `name` parameter with a default value of "Guest". The function should return a greeting message with the provided name or "Guest" if no name is provided.

Expected Output:

```
message = greet_user()
print(message)  # Output: Hello, Guest!
```

Problem 6: Function with a variable number of arguments (variadic function)

Write a function called `sum_numbers` that takes a variable number of arguments and returns the sum of all the arguments.

Expected Output:

```
result = sum_numbers(2, 4, 6, 8, 10)
print(result)  # Output: 30
```

Problem 7: Nested Functions

Write a function called `square` that takes a number `x` as input and returns the square of that number using a nested function called `multiply` to perform the calculation.

Expected Output:

```
result = square(5)
print(result)  # Output: 25
```

IG | @niihaaarrrr | Dodagatta Nihar

Congrats! Day 23 successful ga complete chesnav :) Repu kaluddam.

▼ Day 24

Functions Part - 2

Default Parameters:

```
def greet_user(name="Guest"):
    print(f"Hello, {name}!")

greet_user()           # Output: Hello, Guest!
greet_user("Alice")     # Output: Hello, Alice!
```

Calling Functions Inside Functions:

```
def greet(name):
    return f"Hello, {name}!"

def greet_and_emphasize(name):
    greeting = greet(name)
    return greeting.upper() + "!!!"

result = greet_and_emphasize("Alice")
print(result)  # Output: HELLO, ALICE!!!
```

Scope of Variables:
Global ga declare chesina variables ni program lo nuv ekkadpadte akkad vaadkovach mowa! Ade nuv oka variable ni function lopata declare chesthe nuv daani aa particular function lo maatrame vaadkogalav.

```
global_variable = "I'm global"

def function_with_local_variable():
    local_variable = "I'm local"
    print(global_variable)     # Output: I'm global
    print(local_variable)      # Output: I'm local

function_with_local_variable()
print(global_variable)         # Output: I'm global
# print(local_variable)       # Raises NameError: name 'local_variable' is not def
ined
```

Questions to practice (Day - 24):

IG | @nihaaarrrr | Dodagatta Nihar

Question 1: Simple function with a default parameter
Write a function called `greet_user` that takes an optional `name` parameter with a default value of "Guest". The function should return a greeting message with the provided name or "Guest" if no name is provided.

Expected Output:

```
greet_user()           # Output: Hello, Guest!
greet_user("Alice")     # Output: Hello, Alice!
```

Question 2: Function with multiple parameters
Write a function called `calculate_sum` that takes three parameters `a`, `b`, and `c`, and returns the sum of the three numbers.

Expected Output:

```
result = calculate_sum(5, 10, 15)
print(result)  # Output: 30
```

Question 3: Function calling another function
Write a function called `square` that takes a number `x` as input and returns the square of that number. Then, write a function called `square_and_double` that takes a number `x`, calls the `square` function, and returns twice the square value.

Expected Output:

```
result = square_and_double(5)
print(result)  # Output: 50
```

Question 4: Nested functions and variable scopes
Write a function called `outer_function` that has a local variable `outer_variable` with the value "I'm outer". Inside the `outer_function`, define another function called `inner_function` that has a local variable `inner_variable` with the value "I'm inner". The `inner_function` should print both the `outer_variable` and `inner_variable`. Then, call the `outer_function` and print the `outer_variable` outside the function.

Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
I'm outer
I'm inner
I'm outer
```

Question 5: Function returning multiple values

Write a function called `divide_and_remainder` that takes two numbers `a` and `b` as input and returns their division result and remainder.

Expected Output:

```
result = divide_and_remainder(15, 4)
print(result)  # Output: (3, 3)
```

Question 6: Recursive function

Write a recursive function called `factorial` that takes a positive integer `n` as input and returns its factorial.

Expected Output:

```
result = factorial(5)
print(result)  # Output: 120
```

Congrats! Day 24 successful ga complete chesnav :) Repu kaluddam.

▼ Day 25

Function Arguments

Positional Arguments:

Positional arguments are the most basic type of arguments. They are matched to function parameters based on their order of appearance.

```
def greet(name, age):
    print(f"Hello, {name}! You are {age} years old.")

# Calling the function with positional arguments
greet("Alice", 25)  # Output: Hello, Alice! You are 25 years old.
```

IG | @niihaaarrrr | Dodagatta Nihar

Keyword Arguments:
Keyword arguments are specified with the parameter name followed by the value, separated by an equal sign. These arguments allow you to pass the values to the function in any order.

```
def greet(name, age):
    print(f"Hello, {name}! You are {age} years old.")

# Calling the function with keyword arguments in a different order
greet(age=25, name="Bob")  # Output: Hello, Bob! You are 25 years old.
```

Default Values:
You can provide default values for function parameters. If a value is not passed for that parameter when the function is called, the default value will be used.

```
def greet(name, age=30):
    print(f"Hello, {name}! You are {age} years old.")

# Calling the function without specifying 'age' argument
greet("Alice")  # Output: Hello, Alice! You are 30 years old.

# Calling the function with 'age' argument
greet("Bob", 25)  # Output: Hello, Bob! You are 25 years old.
```

Questions to practice (Day - 25):

Question 1: Positional Arguments

Write a function called `add` that takes two positional arguments `a` and `b`, and returns their sum.

Expected Output:

```
result = add(5, 10)
print(result)  # Output: 15
```

Question 2: Keyword Arguments

Write a function called `greet` that takes two keyword arguments `name` and `age`, and prints a greeting message with the provided values.

Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
greet(name="Alice", age=25)  # Output: Hello, Alice! You are 25 years old.
```

Question 3: Default Values

Write a function called `multiply` that takes two arguments `a` and `b`, with `b` having a default value of 2. The function should return the product of `a` and `b`.

Expected Output:

```
result = multiply(5)
print(result)  # Output: 10
```

Question 4: Mixing Positional and Keyword Arguments

Write a function called `print_info` that takes three positional arguments `name`, `age`, and `country`, and prints the information in a formatted message.

Expected Output:

```
print_info("Alice", 25, "USA")  # Output: Name: Alice, Age: 25, Country: USA
```

Congrats! Day 25 successful ga complete chesnav :) Repu kaluddam.

▼ Day 26

Built-in Functions

`max()` and `min()` - Returns the maximum and minimum values from a sequence.

```
numbers = [10, 5, 8, 15, 3]
max_value = max(numbers)
min_value = min(numbers)
print(max_value)  # Output: 15
print(min_value)  # Output: 3
```

`sum()` - Returns the sum of elements in a sequence.

IG | @niihaaarrrr | Dodagatta Nihar

```
numbers = [1, 2, 3, 4, 5]
total_sum = sum(numbers)
print(total_sum)  # Output: 15
```

`range()` - Generates a sequence of numbers within a specified range.

```
# Generate numbers from 0 to 4
my_range = range(5)
print(list(my_range))  # Output: [0, 1, 2, 3, 4]
```

`sorted()` - Returns a new sorted list from the elements in an iterable.

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6]
sorted_numbers = sorted(numbers)
print(sorted_numbers)  # Output: [1, 1, 2, 3, 4, 5, 6, 9]
```

These are just a few examples of the built-in functions available in Python. There are many more built-in functions that can be utilized for various purposes, and you can find more details in the Python documentation.

Questions to practice (Day - 26):

Question 1: Using `max()` and `min()`

Write a Python program that takes a list of numbers as input and prints the maximum and minimum values from that list.

Expected Output:

```
numbers = [10, 5, 8, 15, 3]

print(max_value)  # Output: 15
print(min_value)  # Output: 3
```

Question 2: Using `sum()`

Write a Python program that takes a list of numbers as input and prints the sum of all the elements in that list.

Expected Output:

IG | @niihaaarrrr | Dodagatta Nihar

```
numbers = [1, 2, 3, 4, 5]

print(total_sum)  # Output: 15
```

Question 3: Using `range()`

Write a Python program that generates a list of even numbers from 2 to 10 using the `range()` function.

Expected Output:

```
# Generate numbers from 2 to 10 with a step of 2

print(list(my_range))  # Output: [2, 4, 6, 8, 10]
```

Question 4: Using `sorted()`

Write a Python program that takes a list of numbers as input, sorts the list in ascending order using the `sorted()` function, and prints the sorted list.

Expected Output:

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6]

print(sorted_numbers)  # Output: [1, 1, 2, 3, 4, 5, 6, 9]
```

Question 5: Using `len()`

Write a Python program that takes a string as input and prints the length of the string using the `len()` function.

Expected Output:

```
text = "Hello, World!"
print(length)  # Output: 13
```

Question 6: Using `abs()`

Write a Python program that takes a number as input and prints its absolute value using the `abs()` function.

Expected Output:

```
number = -10
```

IG | @niihaaarrrr | Dodagatta Nihar

```
print(abs_value)  # Output: 10
```

Question 7: Using `all()` and `any()`

Write a Python program that takes a list of boolean values as input and checks if all values are `True` using the `all()` function. Then, check if at least one value is `True` using the `any()` function.

Expected Output:

```
bool_list = [True, True, False, True]

print(all_true)  # Output: False
print(any_true)  # Output: True
```

Congrats! Day 26 successful ga complete chesnav :) Repu kaluddam.

▼ Day 27

Pattern Printing

Right Triangle Star Pattern

Printing star patterns is a common exercise to practice looping in Python. Here are some simple star patterns you can print:

Right Triangle Star Pattern:
Print a right-angled triangle of stars.

```
def right_triangle(rows):
    for i in range(1, rows + 1):
        print("*" * i)

right_triangle(5)
# Output:
# *
# **
# ***
# ****
# *****
```

Pyramid Star Pattern:
Print a pyramid of stars.

IG | @nihaaarrrr | Dodagatta Nihar

```
def pyramid(rows):
    for i in range(1, rows + 1):
        spaces = " " * (rows - i)
        stars = "*" * (2 * i - 1)
        print(spaces + stars)

pyramid(5)
# Output:
#     *
#    ***
#   *****
#  *******
# *********
```

Hollow Square Star Pattern:
Print a hollow square of stars.

```
def hollow_square(rows):
    for i in range(rows):
        if i == 0 or i == rows - 1:
            print("*" * rows)
        else:
            print("*" + " " * (rows - 2) + "*")

hollow_square(5)
# Output:
# *****
# *   *
# *   *
# *   *
# *****
```

These are just a few examples of star patterns you can create using loops in Python. You can modify the number of rows or add more patterns by adjusting the loop ranges and the number of stars and spaces printed in each row.

Questions to practice (Day - 27):

Question 1: Inverted Right Triangle Star Pattern
Expected Input: Rows: 5
Expected Output:

```
*****
****
```

IG | @niihaaarrrr | Dodagatta Nihar

```
***
**
*
```

Question 2: Half Pyramid Star Pattern

Expected Input: Rows: 5

Expected Output:

```
    *
   **
  ***
 ****
*****
```

Question 3: Hollow Right Triangle Star Pattern

Expected Input: Rows: 5

Expected Output:

```
*
**
* *
*  *
*****
```

Question 4: Full Pyramid Star Pattern

Expected Input: Rows: 5

Expected Output:

```
    *
   ***
  *****
 *******
*********
```

Question 5: Rhombus Star Pattern

Expected Input: Rows: 5

Expected Output:

```
    *
   **
  ***
 ****
*****
```

IG | @niihaaarrrr | Dodagatta Nihar

```
****
 ***
  **
   *
```

Congrats! Day 27 successful ga complete chesnav :) Repu kaluddam.

▼ Day 28

1. Create a variable `name` and assign your name to it. Print a greeting message using the variable.
2. Convert the string `"42"` to an integer and assign it to a variable `num`. Print the value and type of `num`.
3. Define a tuple with three elements: `"cat"`, `"dog"`, and `"rabbit"`. Print the third element.
4. Swap the values of two variables, `a` and `b`, without using a temporary variable.
5. Write a program that takes your age as input and prints a message: `"You are [age] years old."`
6. Calculate the product of two numbers entered by the user and print the result.
7. Format the variables `item = "book"` and `price = 25.99` into a sentence.
8. Write a function that takes an integer as input and returns `"Positive"`, `"Negative"`, or `"Zero"`.
9. Create a program that checks if a user-input number is even or odd.
10. Write a program that determines if a year entered by the user is a leap year.
11. Print the numbers from 1 to 5 using a `for` loop.
12. Write a `while` loop that calculates the sum of numbers from 1 to 20.
13. Print each character of the string `"Hello"` using a loop.
14. Create a list of your favorite fruits and print the second fruit on the list.
15. Add the number 7 to a set. Add the number 7 again and observe the set's behavior.

IG | @nihaaarrrr | Dodagatta Nihar

16. Write a function that takes a list as input and returns a new list without duplicate elements.
17. Create a tuple containing your birth year, birth month, and birth day.
18. Define a function `calculate_average` that takes a list of numbers as input and returns their average.
19. Write a function `power` that takes two arguments, `base` and `exponent`, and calculates `base` raised to the power of `exponent`.
20. Define a function that takes any number of strings as arguments and returns them concatenated.
21. Check if the word `"apple"` is present in the string `"I like apples and oranges"`.
22. Count the occurrences of the letter `"i"` in the string `"Mississippi"`.
23. Reverse the string `"Python"` using slicing.
24. Create a dictionary representing a car with keys `"make"`, `"model"`, and `"year"`. Print the car's model.
25. Add a new car to the dictionary, modify the year of an existing car, and retrieve the make of a specific car.

Congrats! Day 28 successful ga complete chesnav :) Repu kaluddam.

▼ Day 29

1. Create a variable `name` and assign your name to it. Print a message using the variable, such as: "Hello, [name]!"
2. Convert the string `"37"` to an integer and assign it to a variable `num`. Print the value and type of `num`.
3. Define a tuple with four elements: `"apple"`, `"banana"`, `"cherry"`, and `"date"`. Print the second element.
4. Swap the values of two variables, `x` and `y`, without using a temporary variable.
5. Write a program that takes your age as input and prints a sentence like: "You are [age] years old."
6. Calculate the product of two numbers entered by the user and print the result.

IG | @niihaaarrrr | Dodagatta Nihar

7. Format the variables `item = "hat"` and `price = 19.99` into a sentence.
8. Write a function that takes an integer as input and returns whether it's a prime number or not.
9. Create a program that checks if a user-input number is positive, negative, or zero.
10. Write a program that prints the Fibonacci sequence up to a specified number of terms.
11. Print the multiplication table of 7 using nested loops.
12. Print each character of the string `"Python"` on a new line using nested loops.
13. Create a list of your favorite colors and print each color in a sentence: "I like [color]."
14. Add the number 5 to a set. Add the number 5 again and observe the set's behavior.
15. Write a function that takes a list as input and returns a new list with unique elements only.
16. Create a tuple containing your birth year, birth month, and birth day. Print each element separately.
17. Define a function `calculate_average` that takes a list of numbers as input and returns their average.
18. Write a function `factorial` that calculates the factorial of a given number using recursion.
19. Define a function that takes any number of strings as arguments and returns them concatenated.
20. Check if the word `"apple"` is present in the string `"I like apples and oranges."`
21. Count the occurrences of the letter `"t"` in the string `"Testing the text for t's."`
22. Reverse the string `"Python"` using a loop.
23. Create a dictionary representing a book with keys `"title"`, `"author"`, and `"year"`. Print the author's name.
24. Add a new book to the dictionary, modify the title of an existing book, and retrieve the year of a specific book.

IG | @niihaaarrrr | Dodagatta Nihar

25. Print a pattern of right-angled triangles using nested loops:

```
*
**
***
****
*****
```

Congrats! Day 29 successful ga complete chesnav :) Repu kaluddam.

▼ Day 30

1. What is the purpose of a variable in Python?
   a) To store data
   b) To perform calculations
   c) To create loops
   d) To define functions

2. Which of the following data types is immutable in Python?
   a) List
   b) Set
   c) Dictionary
   d) Tuple

3. How do you convert the string `"50"` to an integer in Python?
   a) `int("50")`
   b) `convert_to_int("50")`
   c) `str_to_int("50")`
   d) `integer("50")`

4. Which loop is best suited when the number of iterations is known in advance?
   a) `for` loop
   b) `while` loop
   c) Both are equally suitable
   d) It depends on the situation

5. What is the output of the following code snippet?

```
x = 5
x += 3
print(x)
```

IG | @nilhaaarrrr | Dodagatta Nihar

a) 3
b) 5
c) 8
d) 15

6. How do you take user input in Python?
   a) `input()`
   b) `user_input()`
   c) `get_input()`
   d) `read_input()`

7. Which conditional statement is used to check multiple conditions?
   a) `if`
   b) `else`
   c) `elif`
   d) `then`

8. What does the `range(3, 10)` function generate?
   a) A sequence from 0 to 9
   b) A sequence from 3 to 10
   c) A sequence from 3 to 9
   d) A sequence from 3 to 11

9. What is the output of `print("Hello", end=" ")` followed by `print("World")`?
   a) `Hello World`
   b) `Hello`
   c) `World`
   d) `Hello\\nWorld`

10. Which of the following is used to define a function in Python?
    a) `func`
    b) `def`
    c) `define`
    d) `function`

11. Which data type is best suited to store a collection of unique values?
    a) List
    b) Set
    c) Dictionary
    d) Tuple

IG | @niihaaarrrr | Dodagatta Nihar

12. How do you check if a certain value exists in a list?
    a) `value in list`
    b) `list.contains(value)`
    c) `list.includes(value)`
    d) `list.exists(value)`

13. What will the following code print?

```
for i in range(5):
    print(i * "*")
```

    a) Prints the numbers from 0 to 4
    b) Prints a pattern of asterisks
    c) Prints a pattern of numbers
    d) Generates an error

14. What does the `len()` function return for a dictionary?
    a) The number of keys in the dictionary
    b) The number of values in the dictionary
    c) The sum of the lengths of all keys and values
    d) The total number of items in the dictionary

15. What is the purpose of a `return` statement in a function?
    a) It terminates the function
    b) It outputs a value from the function
    c) It defines a loop
    d) It defines a condition

16. How do you reverse a string `"Python"`?
    a) `"Python"[::-1]`
    b) `reverse("Python")`
    c) `reverse_string("Python")`
    d) `"Python".reverse()`

17. Which of the following is a valid way to add a key-value pair to a dictionary?
    a) `dict.add(key, value)`
    b) `dict[key] = value`
    c) `dict.insert(key, value)`
    d) `dict.append(key, value)`

18. What is the output of `print(3 * 5)`?
    a) 8

IG | @niihaaarrrr | Dodagatta Nihar

b) 15
c) 53
d) `15`

19. What does the following code snippet do?

```
num = 10
while num > 0:
    print(num)
    num -= 2
```

a) Prints even numbers from 10 to 0
b) Prints odd numbers from 10 to 0
c) Prints numbers from 10 to 0 in steps of 2
d) Generates an error

20. Write nested loops to print the following pattern:

```
*
**
***
****
```

a) This pattern cannot be achieved using nested loops.
b) It will print a different pattern.
c) The correct nested loops are included below this question.
d) It depends on the input provided to the loops.

21. Which of the following is NOT a valid variable name in Python?
a) `my_var`
b) `variable_name`
c) `123_variable`
d) `_underscore`

22. What does the `input()` function do in Python?
a) Displays output on the screen
b) Takes user input from the keyboard
c) Converts data types
d) Performs mathematical calculations

23. What will the following code snippet print?

IG | @niihaaarrrr | Dodagatta Nihar

```
for i in range(3):
    for j in range(i+1):
        print("*", end="")
    print()
```

a) Prints a pattern of asterisks: `*`, `**`, `***`
b) Prints a pattern of asterisks: `***`, `**`, `*`
c) Prints a pattern of asterisks: `*`, `*`, `*`
d) Generates an error

24. Which of the following is an example of a mutable data type?
    a) String
    b) Integer
    c) Float
    d) List

25. What is the purpose of the `elif` statement in Python?
    a) It defines a loop
    b) It handles exceptions
    c) It follows the `else` statement
    d) It provides an alternative condition to check

26. What will the following code snippet output?

```
sentence = "Python programming is fun"
words = sentence.split()
print(len(words))
```

27. How do you access the value associated with the key `"age"` in a dictionary `person`?
    a) `person("age")

    b)person["age"]

    c)person.get("age")

    d)person.value("age")`

28. What does the `break` statement do in a loop?
    a) It continues to the next iteration of the loop
    b) It exits the loop immediately

IG | @niihaaarrrr | Dodagatta Nihar

c) It skips the current iteration and moves to the next one

d) It raises an error

29. What is the result of `10 / 3` in Python?
    a) `3.3333`
    b) `3.0`
    c) `3`
    d) `3.333`

30. Write nested loops to print the following pattern:

```
   *
  ***
 *****
*******
```

Congrats! Python 30 Day Reel series successful ga complete chesnav :)
There’s something big coming up, stay tuned for the announcement!

▼ OOP

Assignment on Classes and Objects!

Complete the following code.

```
class Student:
    def __init__(self, name, age):
        self.name = name
        self.age = age
        self.grades = []

    def add_grades(self, grade):
        """
        Add a grade to the list of grades for the student.

        Args:
            grade (float): The grade to be added to the list.
        """

    def calculate_average(self):
        """
        Calculate the average of all grades in the list.

        Returns:
            float: The average grade.
        """
```

IG | @niihaaarrrr | Dodagatta Nihar

```
    def display_student_info(self):
        """
        Display student information, including name, age, and average grade.
        """

# Example usage of the Student class:
student1 = Student("Dodagatta Nihar", 19)
student1.add_grades(90)
student1.add_grades(85)
student1.add_grades(92)

student1.display_student_info()
```

IG | @niihaaarrrr | Dodagatta Nihar