

LOOPS IN PYTHON

1. Print First 10 natural numbers using while loop

Help: [while loop in Python](#)

Expected output:

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

The screenshot shows a web browser window for the CodeChef website. The URL is codechef.com/python-online-compiler. The page has a dark theme. At the top, there are navigation links for Courses, Practice, Compete, Upgrade to Pro, Login, and Sign Up. A banner on the right says "Learning Python? Check out our complete Python roadmap". The main area is titled "Online Python Compiler" and shows a code editor with Python3 selected. The code in the editor is:1
2 i = 1
3 while i <= 10:
4 print(i)
5 i += 1

Below the code editor is a "Run" button. To its right is a text input field labeled "Enter Input here" with placeholder text "If your code takes input, add it in the above box before running.". To the right of the input field is an "Output" section. It displays the status "Status : Successfully executed", execution time "Time: 0.0100 secs", and memory usage "Memory: 8.184 Mb". Under the "Your Output" heading, the numbers 1 through 5 are listed. The bottom of the screen shows a taskbar with various icons and system status information.

2 . Print the following pattern

Write a program to print the following number pattern using a loop.

```
1  
1 2
```

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

A screenshot of a web-based Python online compiler. The code in the editor is:

```
1 for i in range(1,5):
2     for j in range(1,i):
3         print(j,end=' ')
4     print()
```

The 'Run' button is highlighted. Below it is an input field labeled 'Enter Input here'. The 'Output' section shows the status 'Status : Successfully executed' and performance metrics 'Time: 0.0100 secs' and 'Memory: 8.24 Mb'. The 'Your Output' box contains the printed numbers:

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

The browser's address bar shows 'codechef.com/python-online-compiler'. The system tray at the bottom indicates it's 28°C, mostly cloudy, and shows other system icons.

Exercise 3: Calculate the sum of all numbers from 1 to a given number

Write a program to accept a number from a user and calculate the sum of all numbers from 1 to a given number

For example, if the user entered **10** the output should be **55**
(**1+2+3+4+5+6+7+8+9+10**)

Expected Output:

```
Enter number 10
Sum is: 55
```

The screenshot shows a web browser window with the URL `codechef.com/python-online-compiler`. The page title is "Online Python Compiler". The code editor contains the following Python code:

```
1 a=10
2 sum=0
3 for i in range(a+1):
4     sum+=i
5 print(sum)
```

Below the code editor is a "Run" button. To the right of the run button is a "Status" bar with the message "Status : Successfully executed". Underneath the status bar, there are performance metrics: "Time: 0.0200 secs" and "Memory: 8.212 Mb". The "Output" section displays the result "55".

Exercise 4: Write a program to print multiplication table of a given number

Given:

```
num = 2
```

Expected output is:

```
2
4
6
8
10
12
14
16
18
20
```

```
1 a=2
2 c=1
3 for i in range(1,11):
4     c=a*i
5     print(c)
```

Enter Input here

If your code takes input, add it in the above box before running.

Output

Status : Successfully executed

Time: 0.0200 secs | Memory: 8.292 Mb

Your Output

```
4
6
8
10
12
14
16
18
20
```

Exercise 5: Display numbers from a list using loop

Write a program to display only those numbers from a [list](#) that satisfy the following conditions

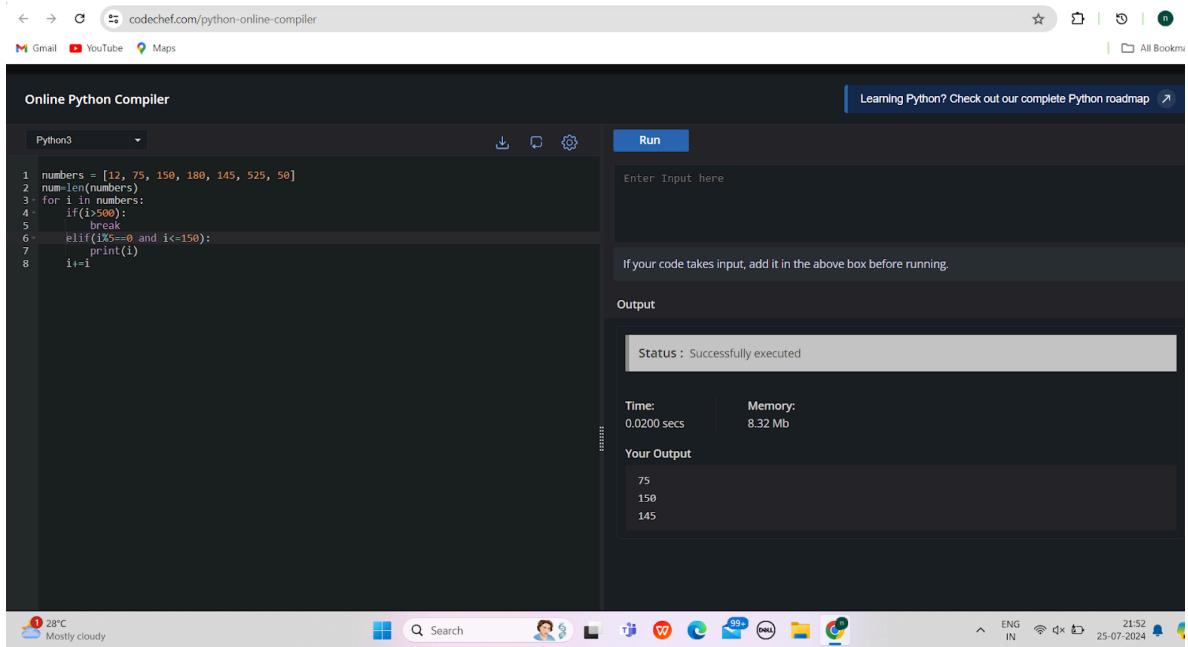
- The number must be divisible by five
- If the number is greater than 150, then skip it and move to the next number
- If the number is greater than 500, then stop the loop

Given:

```
numbers = [12, 75, 150, 180, 145, 525, 50]
```

Expected output:

```
75  
150  
145
```



The screenshot shows a web-based Python compiler interface. The code in the editor is:

```
1 numbers = [12, 75, 150, 180, 145, 525, 50]  
2 num=len(numbers)  
3 for i in numbers:  
4     if(i>=500):  
5         break  
6     elif(i>=0 and i<=150):  
7         print(i)  
8     i+=1
```

The 'Run' button is highlighted. The 'Output' section shows the status 'Successfully executed' and the output '75\n150\n145'. Below the output, system status icons show it's 21:52, 25-07-2024, ENG IN, and a mostly cloudy weather icon.

Exercise 6: Count the total number of digits in a number

Write a program to count the total number of digits in a number using a [while loop](#).

For example, the number is **75869**, so the output should be **5**.

The screenshot shows the CodeChef Online Python Compiler. In the code editor, the following Python 3 code is written:

```
a=876543
count=0
while a!=0:
    a=a//10
    count=count+1
print(count)
```

The 'Run' section contains an input field labeled 'Enter Input here' with placeholder text 'If your code takes input, add it in the above box before running.' Below the input field is an 'Output' section. It displays the status 'Status : Successfully executed' and performance metrics 'Time: 0.0100 secs' and 'Memory: 8.296 Mb'. Under 'Your Output', the number '6' is shown.

Exercise 7: Print the following pattern

Write a program to use `for` loop to print the following reverse number pattern

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

A screenshot of a web browser displaying an online Python compiler. The URL is `codechef.com/python-online-compiler`. The page has a dark theme. On the left, there is a code editor with Python3 code:

```
1 n = 5
2 for i in range(0,n+1):
3     for j in range(n-1,0,-1):
4         print(j,end=" ")
5     print()
```

On the right, there is a "Run" button and an "Output" section. The output shows:

Status : Successfully executed

Time: 0.0100 secs Memory: 8.196 Mb

Your Output

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

The browser's toolbar at the bottom includes icons for search, refresh, and various extensions. The status bar shows the date and time: 25-07-2024 22:06.

Exercise 8: Print list in reverse order using a loop

Given:

```
list1 = [10, 20, 30, 40, 50]
```

Expected output:

```
50
40
30
20
10
```

The screenshot shows a browser window for the CodeChef Online Python Compiler. The URL is codechef.com/python-online-compiler. The code editor contains the following Python script:

```
1 a=[10,20,30,40,50,60]
2 b=reversed(a)
3 for i in b:
4     print(i)
```

The 'Run' button is highlighted in blue. Below it is a text input field labeled 'Enter Input here'. A note says 'If your code takes input, add it in the above box before running.' The 'Output' section shows the status 'Status : Successfully executed'. It includes performance metrics: Time: 0.0200 secs and Memory: 8.22 Mb. The 'Your Output' section displays the reversed list: 60, 50, 40, 30, 20, 10.

Exercise 9: Display numbers from -10 to -1 using for loop

Expected output:

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

The screenshot shows a web-based Python compiler interface. In the code editor, there is a single-line code snippet:

```
a=-10  
for i in range(a,0):  
    print(i)
```

The 'Run' button is highlighted. Below it, an input field says 'Enter Input here' with the note 'If your code takes input, add it in the above box before running.' The 'Output' section shows the status 'Successfully executed' and performance metrics: Time: 0.0300 secs and Memory: 8.252 Mb. The 'Your Output' section displays the numbers from -10 to -1, each on a new line.

Exercise 10: Use else block to display a message “Done” after successful execution of for loop

For example, the following loop will execute without any error.

Given:

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

Expected output:

```
0  
1  
2  
3  
4  
Done!
```

A screenshot of a web-based Python online compiler. The URL in the address bar is `codechef.com/python-online-compiler`. The code editor on the left contains the following Python script:

```
1 for i in range(5):
2     print(i)
3 print("Done!")
```

The right side of the interface shows the execution results. A "Run" button is at the top. Below it is an "Enter Input here" text box with placeholder text: "If your code takes input, add it in the above box before running." An "Output" section follows, displaying the status "Status : Successfully executed". It also shows performance metrics: "Time: 0.0100 secs" and "Memory: 8.232 Mb". Under the "Your Output" heading, the numbers 0, 1, 2, 3, and 4 are listed, followed by the text "Done!". The bottom of the screen shows a Windows taskbar with various icons and system status information.

Exercise 11: Write a program to display all prime numbers within a range

Note: A Prime Number is a number that cannot be made by multiplying other whole numbers. A prime number is a natural number greater than 1 that is not a product of two smaller natural numbers

Examples:

- 6 is not a prime number because it can be made by $2 \times 3 = 6$
- 37 is a prime number because no other whole numbers multiply together to make it.

Given:

```
# range
start = 25
end = 50
```

Expected output:

Prime numbers between 25 and 50 are:

```
29
31
37
41
43
47
```

The screenshot shows the OnlineGDB beta IDE interface. The code editor contains the following Python script:

```
1  ...
2  ...
3  ...
4  ...
5  ...
6  ...
7  ...
8  for i in range(25, 51):
9      is_prime = True
10     for j in range(2, 10):
11         if i % j == 0:
12             is_prime=False
13             break
14     if(is_prime):
15         print(i)
```

The output window below the code editor displays the prime numbers from 25 to 50:

```
29
31
37
41
43
47
```

Exercise 12: Display Fibonacci series up to 10 terms

The Fibonacci Sequence is a series of numbers. The next number is found by adding up the two numbers before it. The first two numbers are 0 and 1.

For example, 0, 1, 1, 2, 3, 5, 8, 13, 21. The next number in this series above is $13+21 = 34$.

Expected output:

Fibonacci sequence:

0 1 1 2 3 5 8 13 21 34

The screenshot shows a Python code editor interface on the CodeChef website. The code is as follows:

```
1 """Exercise 12: Display Fibonacci series up to 10 terms
2 The Fibonacci Sequence is a series of numbers. The next number is found by adding up the
3 two numbers before it. The first two numbers are 0 and 1.
4 For example, 0, 1, 1, 2, 3, 5, 8, 13, 21. The next number in this series above is 13+21 =
5 34.
6
7 Expected output:
8 Fibonacci sequence:
9 0 1 1 2 3 5 8 13 21 34"""
10 a=0
11 b=1
12 for i in range(10):
13     print(a)
14     c=a+b
15     a=b
16     b=c
```

The 'Run' button is highlighted. The 'Output' section shows the generated Fibonacci sequence from 0 to 34. The status bar indicates the code was successfully executed in 0.0200 secs and 8.248 Mb of memory.

Exercise 13: Find the factorial of a given number

Write a program to use the loop to find the factorial of a given number.

The factorial (symbol: !) means to multiply all whole numbers from the chosen number down to 1.

For example: calculate the factorial of 5

5! = 5 × 4 × 3 × 2 × 1 = 120

Expected output:

120

The screenshot shows the OnlineGDB IDE interface. The code in the main.py file is:

```
1 a=1
2 c=int(input())
3 for i in range(1,c+1):
4     a*=i
5 print(a)
```

The input field contains the value 5. The output window shows the program finished with exit code 0 and the output 120.

Exercise 14: Reverse a given integer number

Given:

76542

Expected output:

24567

The screenshot shows the OnlineGDB IDE interface. The code in the main.py file is:

```
1 a=765432
2 print("Before reversed")
3 print(a)
4 print("After reversed")
5 for i in range(6):
6     b=a%10
7     print(b,end="")
8     a=a//10
```

The output window shows the reversed number 24567.

Exercise 15: Use a loop to display elements from a given list present at odd index positions

Given:

```
my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
```

Note: [list](#) index always starts at 0

Expected output:

```
20 40 60 80 100
```

The screenshot shows the OnlineGDB beta IDE interface. The code editor contains the following Python script:

```
1 """Exercise 15: Use a loop to display elements from a given list present at odd index positions
2
3 Given:
4 my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
5 Note: list index always starts at 0
6
7 Expected output:
8
9 20 40 60 80 100"""
10 my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
11 del(my_list)
12 for i in range(d):
13     if(i%2==0):
14         e=my_list[i]
15         print(e,end=' ')
```

The terminal window below shows the output of the program:

```
20 40 60 80 100
...Program finished with exit code 0
Press ENTER to exit console.
```

The status bar at the bottom right indicates the date and time as 26-07-2024 10:57.

Exercise 16: Calculate the cube of all numbers from 1 to a given number

Write a program to print the cube of all numbers from 1 to a given number

Given:

input_number = 6

Expected output:

```
Current Number is : 1 and the cube is 1
Current Number is : 2 and the cube is 8
Current Number is : 3 and the cube is 27
Current Number is : 4 and the cube is 64
Current Number is : 5 and the cube is 125
Current Number is : 6 and the cube is 216
```

The screenshot shows a web-based IDE interface for OnlineGDB. On the left, there's a sidebar with links for IDE, My Projects, Classroom, Learn Programming, Programming Questions, Jobs, Sign Up, and Login. The main workspace has tabs for 'main.py' and 'Python for loop'. The code in 'main.py' is:

```
1 inp_number=6
2 for i in range(1,inp_number+1):
3     a=i*i
4     print("Current Number is:",i,"and the cube is",a)
```

Below the code, there's an 'input' section containing the output of the program:

```
Current Number is: 1 and the cube is 1
Current Number is: 2 and the cube is 8
Current Number is: 3 and the cube is 27
Current Number is: 4 and the cube is 64
Current Number is: 5 and the cube is 125
Current Number is: 6 and the cube is 216
```

The status bar at the bottom shows system information like battery level (BSE smicap +105%), network, and date/time (26-07-2024).

Exercise 17: Find the sum of the series upto n terms

Write a program to calculate the sum of series up to n term. For example, if n = 5 the series will become $2 + 22 + 222 + 2222 + 22222 = 24690$

Given:

```
# number of terms
```

```
n = 5
```

Expected output:

```
24690
```

The screenshot shows a browser window with the OnlineGDB IDE interface. The code in 'main.py' is:

```
n = 5
# first number of sequence
start = 2
sum_seq = 0

# run loop n times
for i in range(0, n):
    print(start, end=" + ")
    sum_seq += start
    # calculate the next term
    start = start * 10 + 2
print("\nSum of above series is:", sum_seq)
```

The output window shows the execution results:

```
2+22+222+2222+22222+
Sum of above series is: 24690
```

At the bottom, it says "Program finished with exit code 0" and "Press ENTER to exit console."

Exercise 18: Print the following pattern

Write a program to print the following start pattern using the `for` loop

A diamond-shaped pattern of stars, where each row has an odd number of stars. The pattern is symmetrical and consists of 11 rows of stars.

| | | | | |
|---|---|---|---|---|
| * | | | | |
| * | * | | | |
| * | * | * | | |
| * | * | * | * | |
| * | * | * | * | * |
| * | * | * | * | |
| * | * | * | | |
| * | * | | | |
| * | | | | |

The screenshot shows a web-based IDE interface for OnlineGDB. The left sidebar has links for IDE, My Projects, Classroom, Learn Programming, Programming Questions, Jobs, Sign Up, and Login. A banner for "Learn Python with KodeKloud" is visible. The main area shows a code editor with a Python script named `main.py`. The code prints a diamond pattern of asterisks. The output window below shows the resulting pattern.

```
1 '''Exercise 18: Print the following pattern
2 Write a program to print the following start pattern using the for loop
3
4 *
5 **
6 ***
7 ****
8 *****
9 *****
10 ****
11 ***
12 **
13 *
14 a=5
15 for i in range(a):
16     for j in range(0,i+1):
17         print("*",end=' ')
18     print("")
19 for k in range(a+1,0,-1):
20     for l in range(k):
21         print("*",end=' ')
22     print("")
```

input

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

E-BOX

FUNCTIONS IN PYTHON

The screenshot shows a Python code editor interface. On the left, there's a sidebar with numbered steps (1-11) and sections like 'Functional Specifications', 'Input and Output Format', 'Sample Input and Output', and 'Problem Requirements'. Step 1 is expanded, showing a function definition. Step 11 contains sample input and output. The main area is titled 'Program' and shows a code editor with Python3 selected. The code reads multiple user names from input and prints them. A status bar at the bottom indicates the code was run at 00:03:44:01.

```
1 def greet(*args):
2     a=int(input())
3     print("Enter",a,"names")
4     name=[]
5     for i in range(a):
6         b=input()
7         name.append(b)
8     for i in name:
9         print("Hello",i)
10    print("Enter the number of users")
11    greet()
```

This screenshot shows another Python code editor interface. The sidebar has steps 1-14, sections for 'Functional Specifications', 'Input Format', 'Sample Input and Output', and 'Problem Requirements'. Step 14 is expanded, showing a function definition for arithmetic operations. The main area is titled 'Program' with Python3 selected. The code defines a function to perform addition, subtraction, multiplication, and division. It handles integer division for matching sample output. A status bar at the bottom indicates the code was run at 00:03:43:57.

```
1 def arithmeticOperation(a1,a2):
2     print("Addition : " + str(a1 + a2))
3     print("Subtraction : " + str(a1 - a2))
4     print("Multiplication : " + str(a1 * a2))
5     if a2 != 0:
6         division = a1 // a2 # Using integer division to match sample output
7         print("Division : " + str(division))
8     else:
9         division = 'Undefined'
10    print("Enter a")
11    a1 = int(input())
12    print("Enter b")
13    a2 = int(input())
14    arithmeticOperation(a1,a2)
```

Display Image HTML EBOX - Yahoo India Search Results Course

pro.e-box.co.in/attempt/203423

Gmail YouTube Maps

Python Functions / Assess

00:03:43:52 Review & Finish BOYAPATI NAGALAKSHMI SRAVANI

Problem

Displaying user inputs using multiple return values

Write a program to display the user entered details and return multiple values.

Functions Specifications:

```
def multiVarFunc()
```

Input Format:
First line of input consists of String which is department name.
Second line of input consists of integer input which is the total number of students
Third line of input consists of integer input which is the total number of faculty.

Output Format:
Display the user inputs.

Sample Input and Output:
Enter department name:
CSE
Enter the number of total students:
60
Enter the number of total faculties:
15

Program

Python3 Submit Previous Submission

```
1 def multiVarFunc(a,b,c):  
2     print("Details:")  
3     print("Department:{a}")  
4     print("Total students:{b}")  
5     print("Total faculties:{c}")  
6     print("Enter department name:")  
7     a=input()  
8     print("Enter the number of total students:")  
9     b=int(input())  
10    print("Enter the number of total faculties:")  
11    c=int(input())  
12    multiVarFunc(a,b,c)
```

Command line argument

Watchlist Ideas

Search

ENG IN 16:24 26-07-2024

Display Image HTML EBOX - Yahoo India Search Results Course

pro.e-box.co.in/attempt/203423

Gmail YouTube Maps

Python Functions / Assess

00:03:43:48 Review & Finish BOYAPATI NAGALAKSHMI SRAVANI

Problem

Swapping using multiple return values

Write a program to swap two numbers using third variable.

Input format :
Input consists of two integers.

Output format :
Output consists of two integers which are swapped.

Sample Input and Output :
[All text of bold corresponds to input and the rest corresponds to output]
Enter x:
5
Enter y:
2
Before swapping **5** **2**
After swapping **2** **5**

Program

Python3 Submit Previous Submission

```
1 print("Enter x:")  
2 x=int(input())  
3 print("Enter y:")  
4 y=int(input())  
5 print("Before swapping {x} {y}")  
6 a=x  
7 x=y  
8 y=a  
9 print("After swapping {x} {y}")
```

Command line argument

Watchlist Ideas

Search

ENG IN 16:26 26-07-2024

