

21. Write a program to print the first n perfect numbers. (Hint Perfect number means a positive integer that is equal to the sum of its proper divisors)

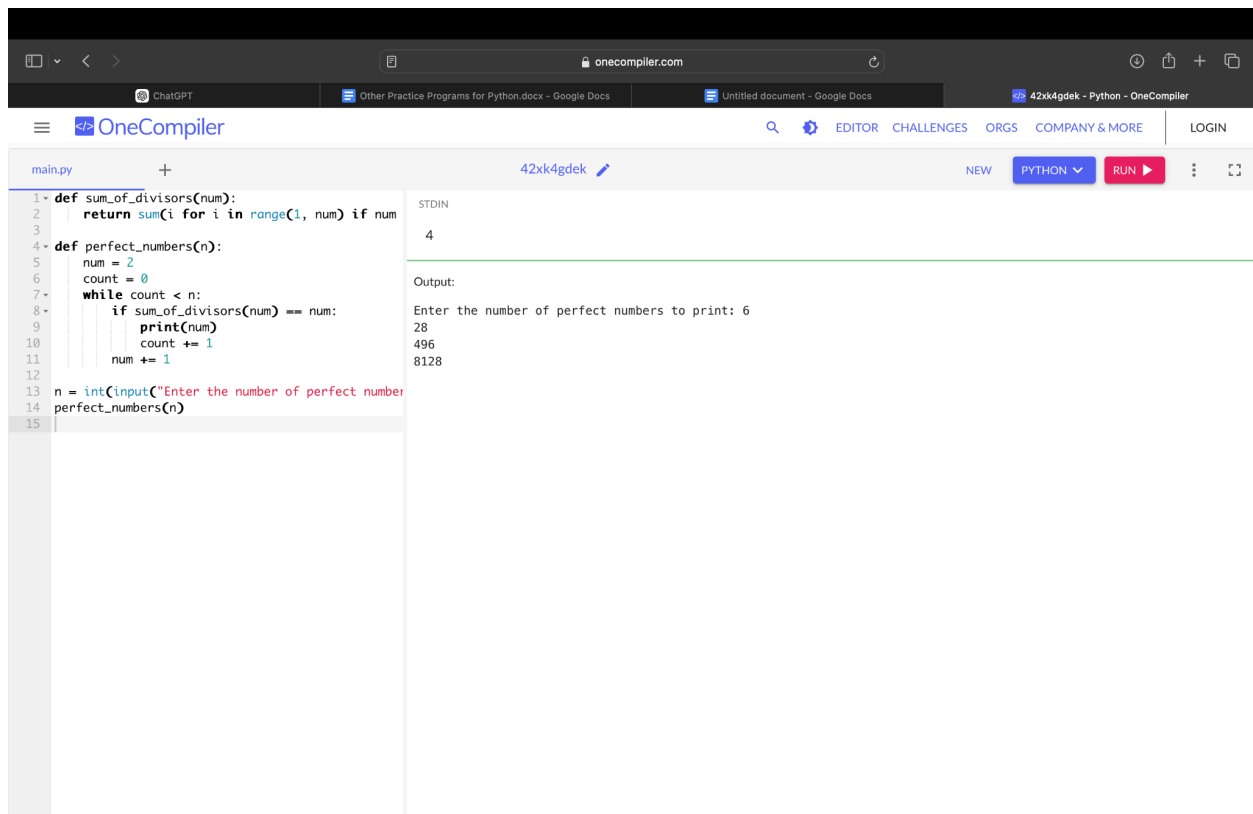
PROGRAM:

```
def sum_of_divisors(num):  
    return sum(i for i in range(1, num) if num % i == 0)
```

```
def perfect_numbers(n):  
    num = 2  
    count = 0  
    while count < n:  
        if sum_of_divisors(num) == num:  
            print(num)  
            count += 1  
        num += 1
```

```
n = int(input("Enter the number of perfect numbers to print: "))  
perfect_numbers(n)
```

OUTPUT:



```
main.py + 42xk4gdek NEW PYTHON RUN
```

```
1 def sum_of_divisors(num):  
2     return sum(i for i in range(1, num) if num % i == 0)  
3  
4 def perfect_numbers(n):  
5     num = 2  
6     count = 0  
7     while count < n:  
8         if sum_of_divisors(num) == num:  
9             print(num)  
10            count += 1  
11            num += 1  
12  
13 n = int(input("Enter the number of perfect number  
14 perfect_numbers(n)  
15
```

STDIN

```
4
```

Output:

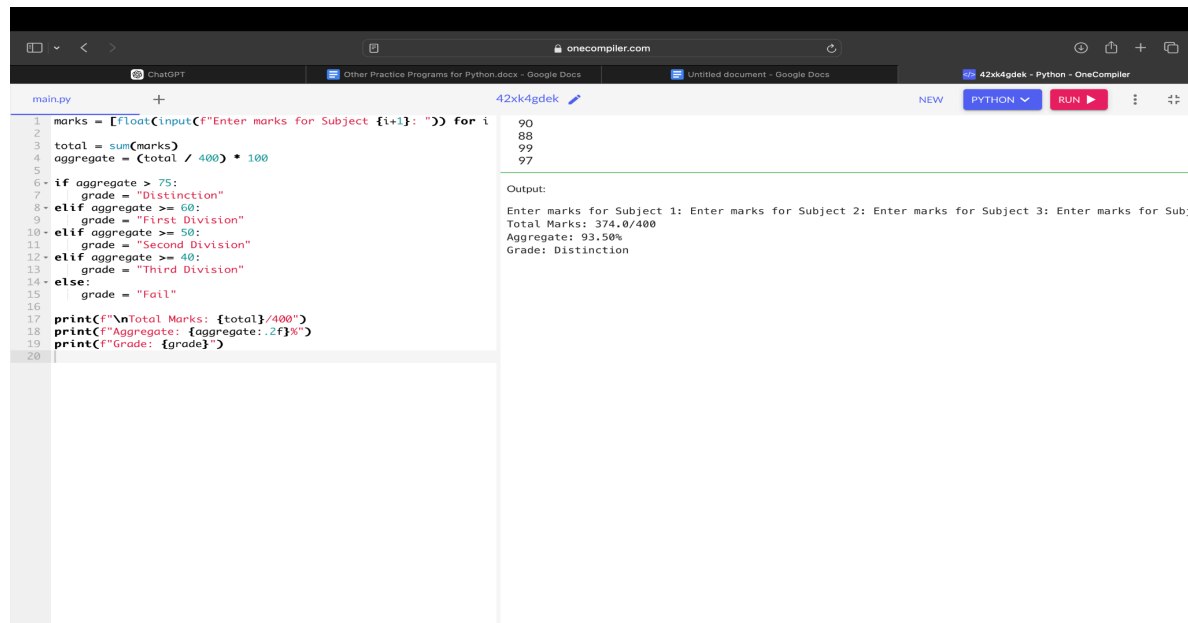
```
Enter the number of perfect numbers to print: 6  
28  
496  
8128
```

22. Write a program to enter the marks of a student in four subjects. Then calculate the total and aggregate, display the grade obtained by the student. If the student scores an aggregate greater than 75%, then the grade is Distinction. If aggregate is $60 \geq$ and < 75 , then the grade is First Division. If aggregate is $50 \geq$ and < 60 , then the grade is Second Division. If aggregate is $40 \geq$ and < 50 , then the grade is Third Division. Else the grade is Fail.

PROGRAM:

```
marks = [float(input(f"Enter marks for Subject {i+1}: ")) for i in range(4)]
total = sum(marks)
aggregate = (total / 400) * 100
if aggregate > 75:
    grade = "Distinction"
elif aggregate >= 60:
    grade = "First Division"
elif aggregate >= 50:
    grade = "Second Division"
elif aggregate >= 40:
    grade = "Third Division"
else:
    grade = "Fail"
print(f"\nTotal Marks: {total}/400")
print(f"Aggregate: {aggregate:.2f}%")
print(f"Grade: {grade}")
```

OUTPUT:



The screenshot shows a web browser window with the URL `onecompiler.com`. The browser has several tabs open, including 'ChatGPT', 'Other Practice Programs for Python.docx - Google Docs', 'Untitled document - Google Docs', and '42xk4gdek - Python - OneCompiler'. The '42xk4gdek - Python - OneCompiler' tab is active, displaying a Python program in a text editor. The code is as follows:

```
1 marks = [float(input(f"Enter marks for Subject {i+1}: ")) for i in range(4)]
2
3 total = sum(marks)
4 aggregate = (total / 400) * 100
5
6 if aggregate > 75:
7     grade = "Distinction"
8 elif aggregate >= 60:
9     grade = "First Division"
10 elif aggregate >= 50:
11     grade = "Second Division"
12 elif aggregate >= 40:
13     grade = "Third Division"
14 else:
15     grade = "Fail"
16
17 print(f"\nTotal Marks: {total}/400")
18 print(f"Aggregate: {aggregate:.2f}%")
19 print(f"Grade: {grade}")
20
```

On the right side of the editor, there is a 'NEW' button, a 'PYTHON' dropdown menu, and a 'RUN' button. Below the code editor, the output of the program is displayed:

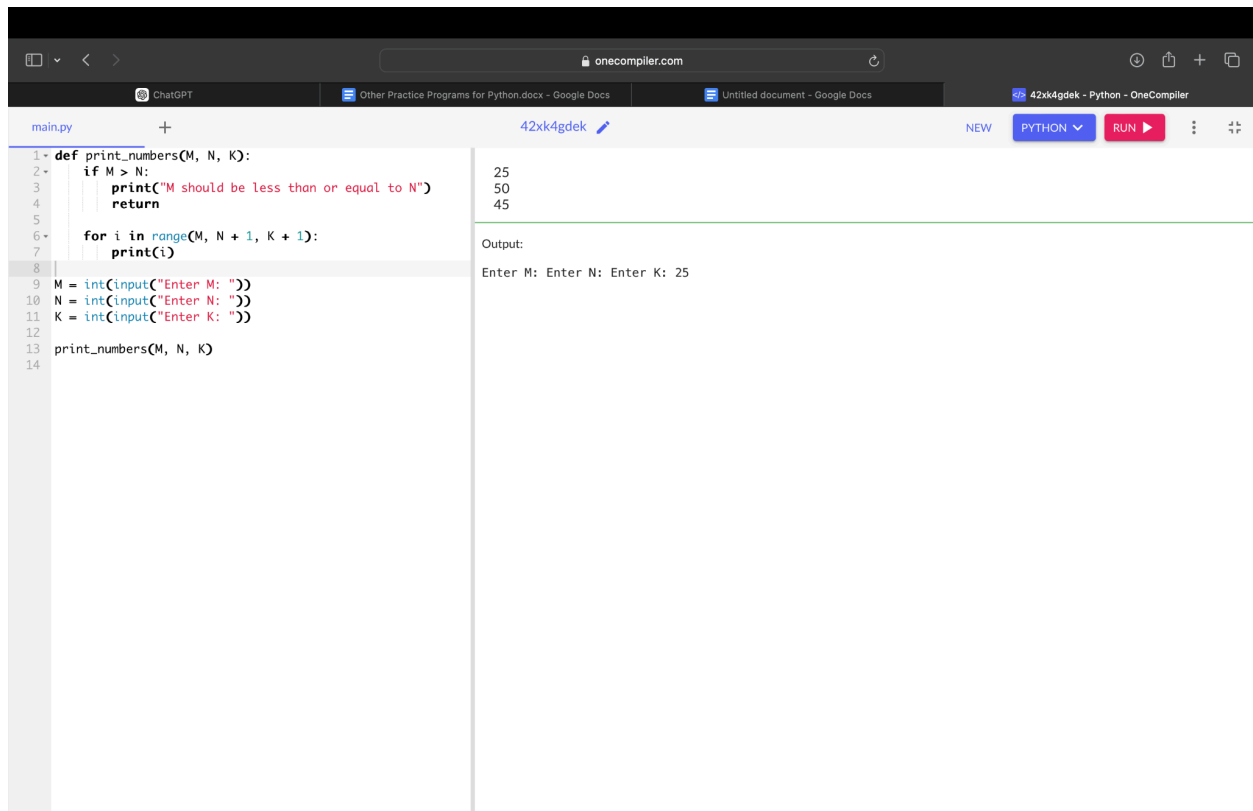
```
Output:
Enter marks for Subject 1: Enter marks for Subject 2: Enter marks for Subject 3: Enter marks for Subj
Total Marks: 374.0/400
Aggregate: 93.50%
Grade: Distinction
```

23. Write a program to print the numbers from M to N by skipping K numbers in between?

PROGRAM:

```
def print_numbers(M, N, K):  
    if M > N:  
        print("M should be less than or equal to N")  
        return  
  
    for i in range(M, N + 1, K + 1):  
        print(i)  
  
M = int(input("Enter M: "))  
N = int(input("Enter N: "))  
K = int(input("Enter K: "))  
  
print_numbers(M, N, K)
```

OUTPUT:



The screenshot shows a web browser window with the URL `onecompiler.com`. The browser has several tabs open, including "ChatGPT", "Other Practice Programs for Python.docx - Google Docs", "Untitled document - Google Docs", and "42xx4gdek - Python - OneCompiler". The "42xx4gdek" tab is active, showing a Python program in a text editor. The program is as follows:

```
1 def print_numbers(M, N, K):  
2     if M > N:  
3         print("M should be less than or equal to N")  
4         return  
5  
6     for i in range(M, N + 1, K + 1):  
7         print(i)  
8  
9 M = int(input("Enter M: "))  
10 N = int(input("Enter N: "))  
11 K = int(input("Enter K: "))  
12  
13 print_numbers(M, N, K)  
14
```

On the right side of the editor, there is a "NEW" button, a "PYTHON" dropdown menu, and a "RUN" button. Below these, the output of the program is displayed:

```
Output:  
Enter M: Enter N: Enter K: 25  
25  
50  
45
```

24. Write a program for matrix addition?

PROGRAM:

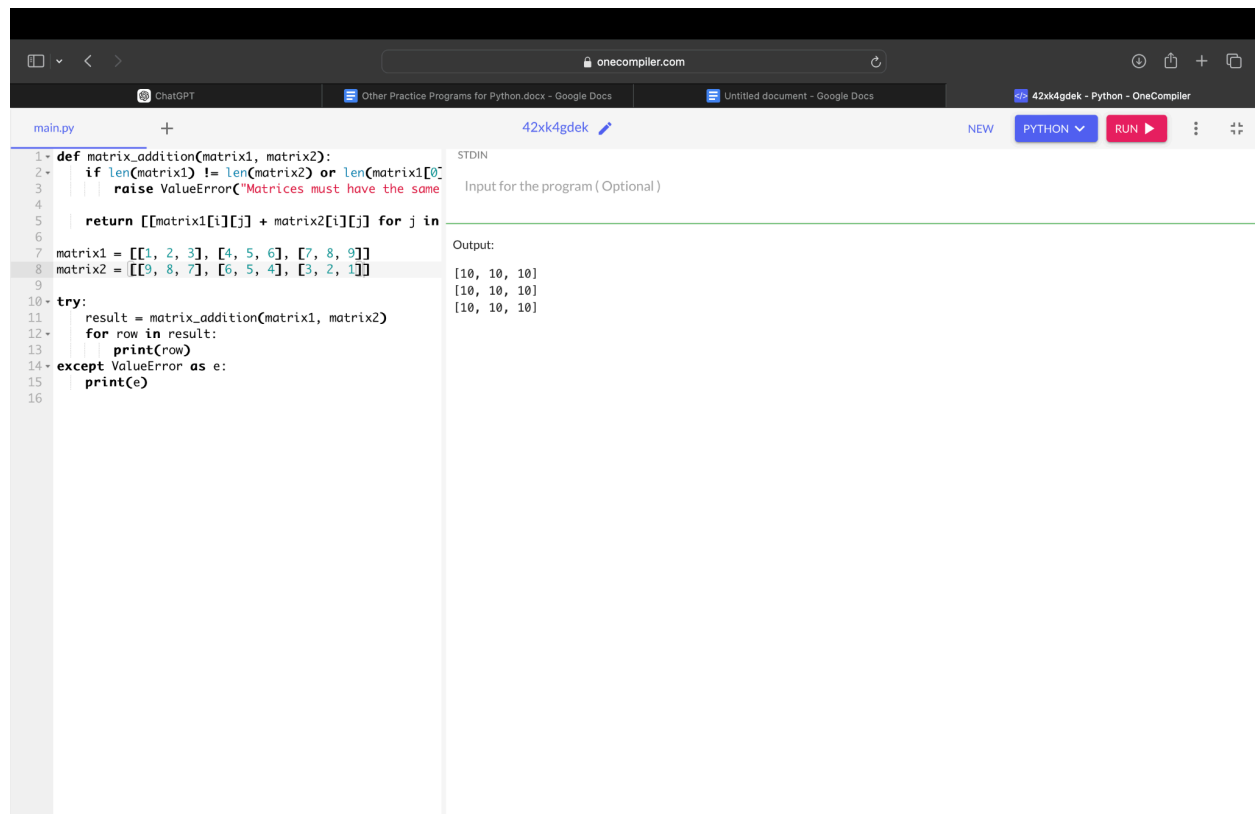
```
def matrix_addition(matrix1, matrix2):
    if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]):
        raise ValueError("Matrices must have the same dimensions.")

    return [[matrix1[i][j] + matrix2[i][j] for j in range(len(matrix1[0]))] for i in
range(len(matrix1))]
```

```
matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
matrix2 = [[9, 8, 7], [6, 5, 4], [3, 2, 1]]
```

```
try:
    result = matrix_addition(matrix1, matrix2)
    for row in result:
        print(row)
except ValueError as e:
    print(e)
```

OUTPUT:



The screenshot shows the OneCompiler online Python IDE. The editor displays the following Python code:

```
1- def matrix_addition(matrix1, matrix2):
2-     if len(matrix1) != len(matrix2) or len(matrix1[0]) != len(matrix2[0]):
3-         raise ValueError("Matrices must have the same dimensions.")
4-
5-     return [[matrix1[i][j] + matrix2[i][j] for j in range(len(matrix1[0]))] for i in
6-             range(len(matrix1))]
7-
8- matrix1 = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]
9- matrix2 = [[9, 8, 7], [6, 5, 4], [3, 2, 1]]
10-
11- try:
12-     result = matrix_addition(matrix1, matrix2)
13-     for row in result:
14-         print(row)
15- except ValueError as e:
16-     print(e)
```

The output window on the right shows the result of the program execution:

```
Output:
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]
```

25. Write a program to calculate tax given the following conditions:
- a. If income is less than or equal to 1,50,000 then no tax
 - b. If taxable income is 1,50,001 – 3,00,000 the charge 10% tax
 - c. If taxable income is 3,00,001 – 5,00,000 the charge 20% tax
 - d. If taxable income is above 5,00,001 then charge 30% tax

PROGRAM:

```
def calculate_tax(income):  
    if income <= 150000:  
        tax = 0  
    elif income <= 300000:  
        tax = (income - 150000) * 0.10  
    elif income <= 500000:  
        tax = (300000 - 150000) * 0.10 + (income - 300000) * 0.20  
    else:  
        tax = (300000 - 150000) * 0.10 + (500000 - 300000) * 0.20 + (income - 500000) *  
0.30  
  
    return tax  
income = float(input("Enter your income: "))  
tax = calculate_tax(income)  
  
print(f"The tax on your income of {income} is: {tax:.2f}")
```

OUTPUT:

The screenshot shows a web-based Python IDE (OneCompiler) with the following code in a file named `main.py`:

```
1- def calculate_tax(income):  
2-     if income <= 150000:  
3-         tax = 0  
4-     elif income <= 300000:  
5-         tax = (income - 150000) * 0.10  
6-     elif income <= 500000:  
7-         tax = (300000 - 150000) * 0.10 + (income - 300000) * 0.20  
8-     else:  
9-         tax = (300000 - 150000) * 0.10 + (500000 - 300000) * 0.20 + (income - 500000) * 0.30  
10-  
11-     return tax  
12-  
13- income = float(input("Enter your income: "))  
14- tax = calculate_tax(income)  
15-  
16- print(f"The tax on your income of {income} is: {tax:.2f}")  
17-
```

The output window shows the following:

STDIN
40.0

Output:
Enter your income: The tax on your income of 40.0 is: 0.00

26. Write a program that would sort a list of names in alphabetical order Ascending or Descending, choice from the user?

PROGRAM:

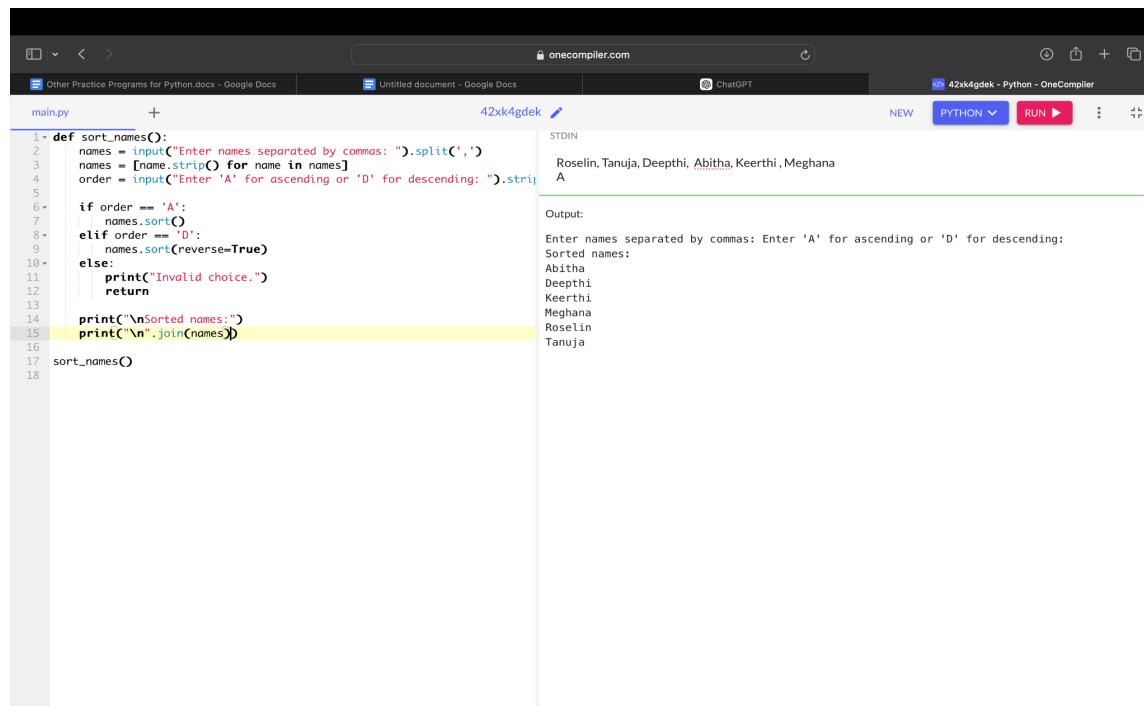
```
def sort_names():
    names = input("Enter names separated by commas: ").split(',')
    names = [name.strip() for name in names]
    order = input("Enter 'A' for ascending or 'D' for descending: ").strip().upper()

    if order == 'A':
        names.sort()
    elif order == 'D':
        names.sort(reverse=True)
    else:
        print("Invalid choice.")
        return

    print("\nSorted names:")
    print("\n".join(names))
```

sort_names()

OUTPUT:



The screenshot shows a web browser window with the OneCompiler website. The code editor on the left contains the Python program for sorting names. The output panel on the right shows the execution results. The user entered the names 'Roselin, Tanuja, Deepthi, Abitha, Keerthi, Meghana' and chose 'A' for ascending order. The output shows the sorted names: Abitha, Deepthi, Keerthi, Meghana, Roselin, and Tanuja.

```
1 def sort_names():
2     names = input("Enter names separated by commas: ").split(',')
3     names = [name.strip() for name in names]
4     order = input("Enter 'A' for ascending or 'D' for descending: ").strip().upper()
5
6     if order == 'A':
7         names.sort()
8     elif order == 'D':
9         names.sort(reverse=True)
10    else:
11        print("Invalid choice.")
12        return
13
14    print("\nSorted names:")
15    print("\n".join(names))
16
17 sort_names()
18
```

STDIN

```
Roselin, Tanuja, Deepthi, Abitha, Keerthi, Meghana
A
```

Output:

```
Enter names separated by commas: Enter 'A' for ascending or 'D' for descending:
Sorted names:
Abitha
Deepthi
Keerthi
Meghana
Roselin
Tanuja
```

27. Write a program for matrix multiplication?

PROGRAM:

```
def multiply_matrices(A, B):
    if len(A[0]) != len(B):
        raise ValueError("Incompatible matrices for multiplication")

    result = [[0] * len(B[0]) for _ in range(len(A))]

    for i in range(len(A)):
        for j in range(len(B[0])):
            result[i][j] = sum(A[i][k] * B[k][j] for k in range(len(B)))

    return result
```

A = [[1, 2, 3], [4, 5, 6]]

B = [[7, 8], [9, 10], [11, 12]]

try:

result = multiply_matrices(A, B)

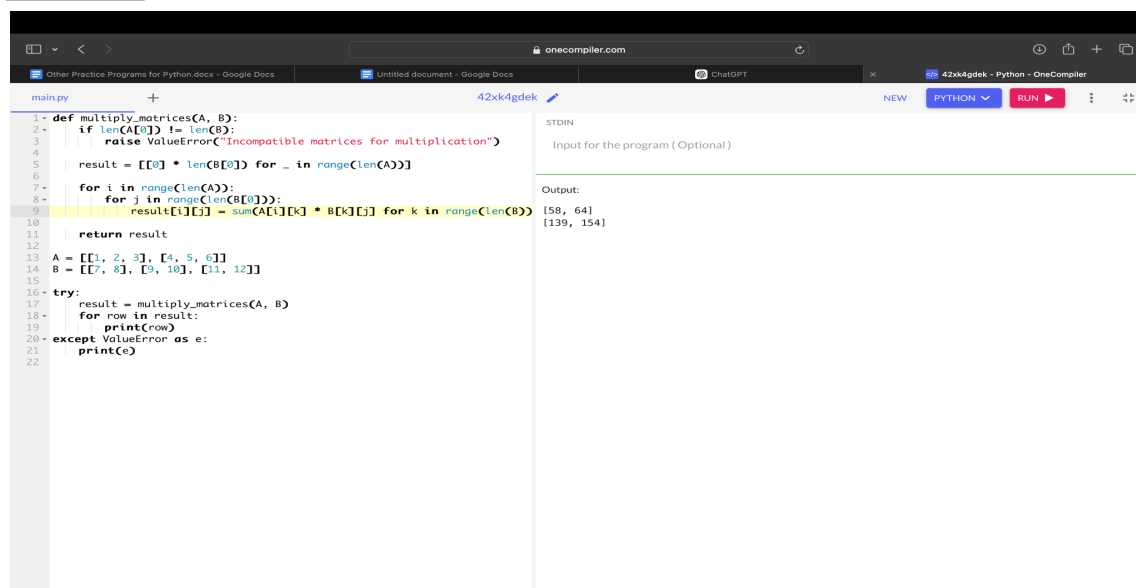
for row in result:

print(row)

except ValueError as e:

print(e)

OUTPUT:



The screenshot shows a web-based Python IDE interface. The code editor on the left contains the following Python code:

```
1- def multiply_matrices(A, B):
2-     if len(A[0]) != len(B):
3-         raise ValueError("Incompatible matrices for multiplication")
4-
5-     result = [[0] * len(B[0]) for _ in range(len(A))]
6-
7-     for i in range(len(A)):
8-         for j in range(len(B[0])):
9-             result[i][j] = sum(A[i][k] * B[k][j] for k in range(len(B)))
10-
11-     return result
12-
13- A = [[1, 2, 3], [4, 5, 6]]
14- B = [[7, 8], [9, 10], [11, 12]]
15-
16- try:
17-     result = multiply_matrices(A, B)
18-     for row in result:
19-         print(row)
20- except ValueError as e:
21-     print(e)
22-
```

The output panel on the right shows the following output:

```
Output:
[58, 64]
[139, 154]
```

29. Write a program to print the following pattern

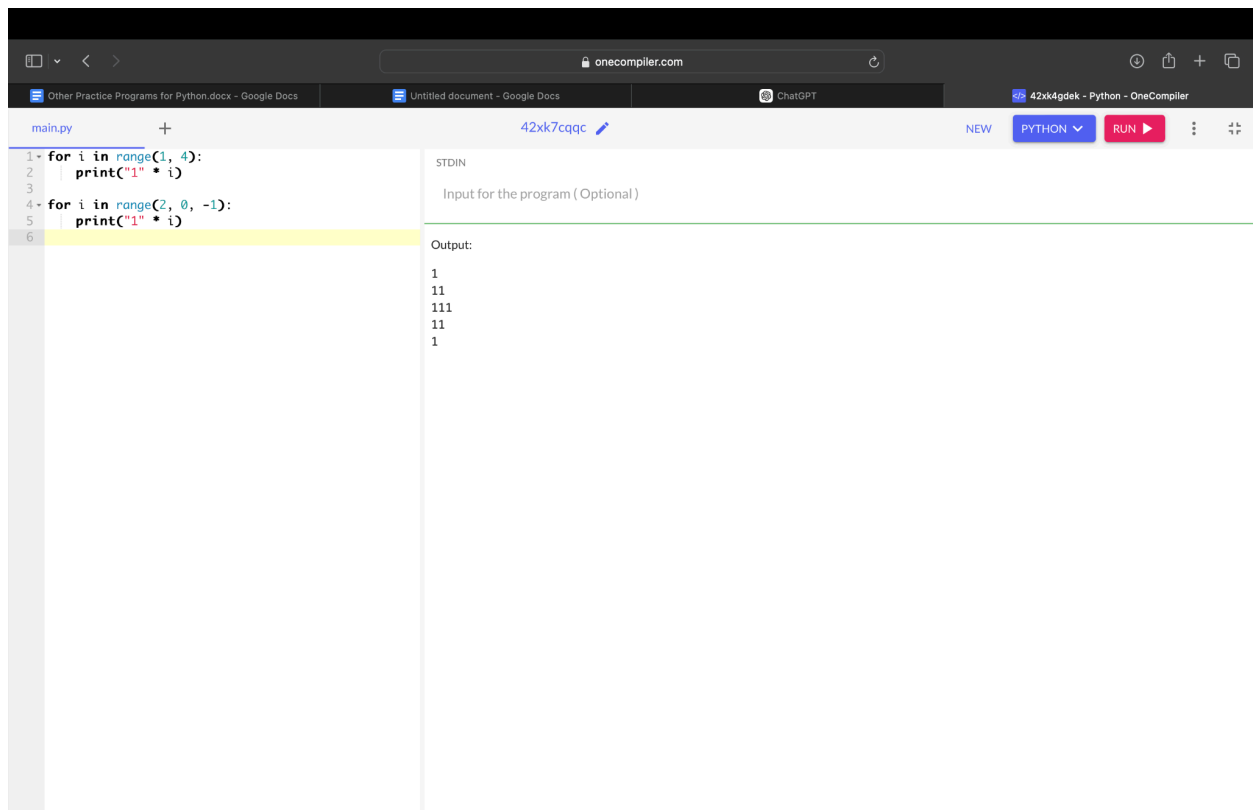
```
1
11
111
11
1
```

PROGRAM:

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

```
for i in range(2, 0, -1):
    print("1" * i)
```

OUTPUT:



The screenshot shows a web browser window with the URL `onecompiler.com`. The browser's address bar and tabs are visible at the top. The main content area displays a Python program in a code editor. The code consists of two loops: the first loop prints the number of '1's from 1 to 3, and the second loop prints the number of '1's from 2 down to 1. The output of the program is shown in a separate pane on the right, displaying the pattern of '1's as specified in the problem statement.

```
main.py + 42xk7cqqc 42xk4gdek - Python - OneCompiler
1 for i in range(1, 4):
2     print("1" * i)
3
4 for i in range(2, 0, -1):
5     print("1" * i)
6
```

STDIN

Input for the program (Optional)

Output:

```
1
11
111
11
1
```

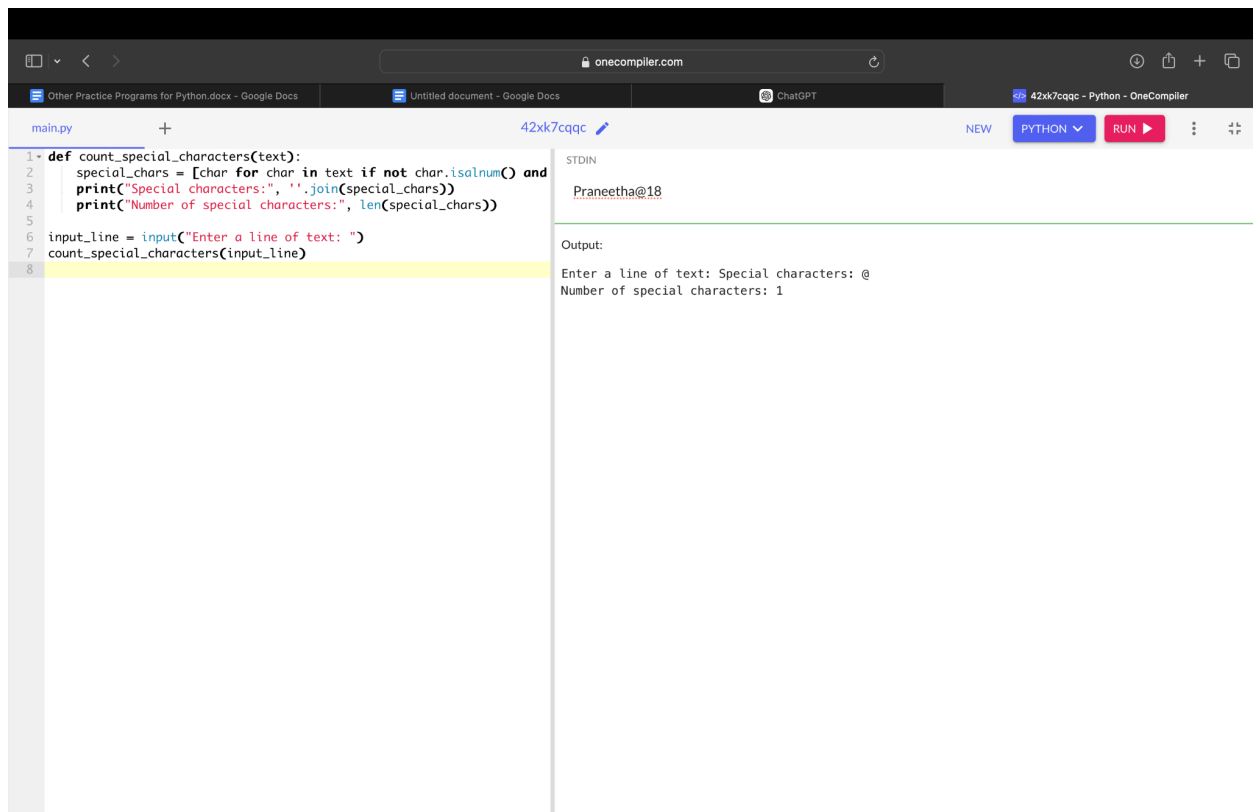

30. Write a program to print the special characters separately and print number of Special characters in the line?

PROGRAM:

```
def count_special_characters(text):  
    special_chars = [char for char in text if not char.isalnum() and not char.isspace()]  
    print("Special characters:", ''.join(special_chars))  
    print("Number of special characters:", len(special_chars))
```

```
input_line = input("Enter a line of text: ")  
count_special_characters(input_line)
```

OUTPUT:



The screenshot shows a web browser window with the URL `onecompiler.com`. The browser's address bar and tabs are visible at the top. The main content area is divided into two panels. The left panel, titled `main.py`, contains the following Python code:

```
1 def count_special_characters(text):  
2     special_chars = [char for char in text if not char.isalnum() and  
3     print("Special characters:", ''.join(special_chars))  
4     print("Number of special characters:", len(special_chars))  
5  
6 input_line = input("Enter a line of text: ")  
7 count_special_characters(input_line)  
8
```

The right panel, titled `STDIN`, shows the input `Praneetha@18`. Below this, the `Output:` section displays the program's execution results:

```
Enter a line of text: Special characters: @  
Number of special characters: 1
```