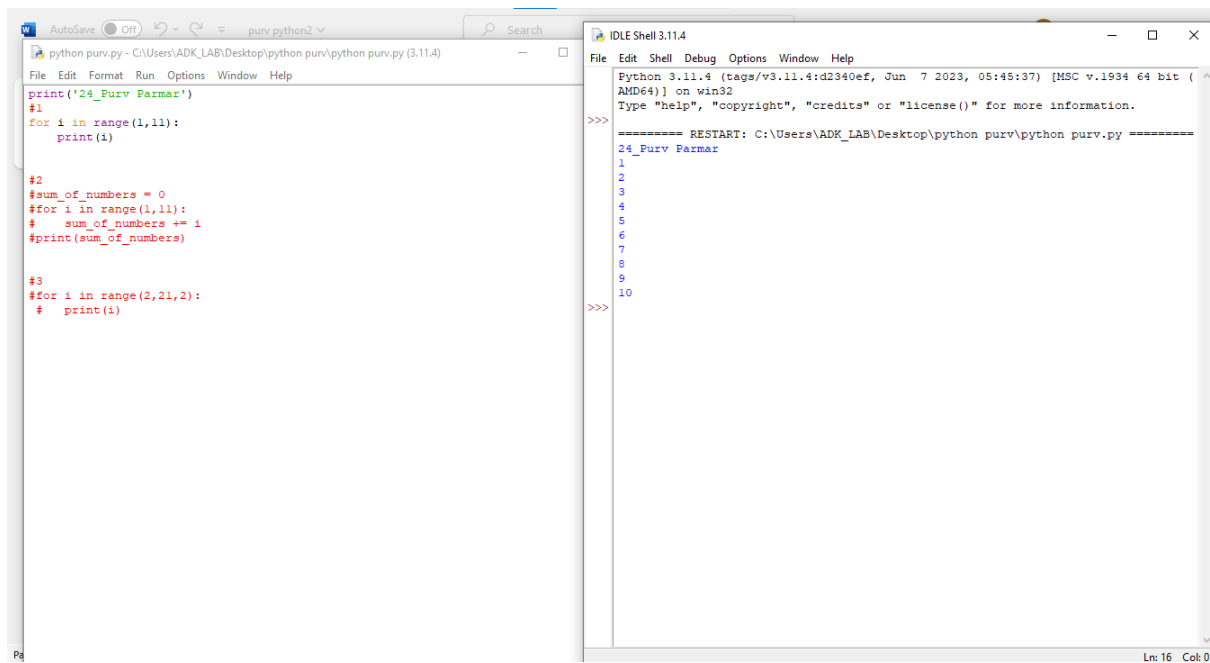


1. Printing the numbers from 1 to 10:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with three sections of code. The first section prints the name '24_Purv Parmar'. The second section uses a for loop to print numbers from 1 to 10. The third section uses a for loop to print numbers from 2 to 2. The right window shows the output of the script, which is the name '24_Purv Parmar' followed by the numbers 1 through 10, each on a new line.

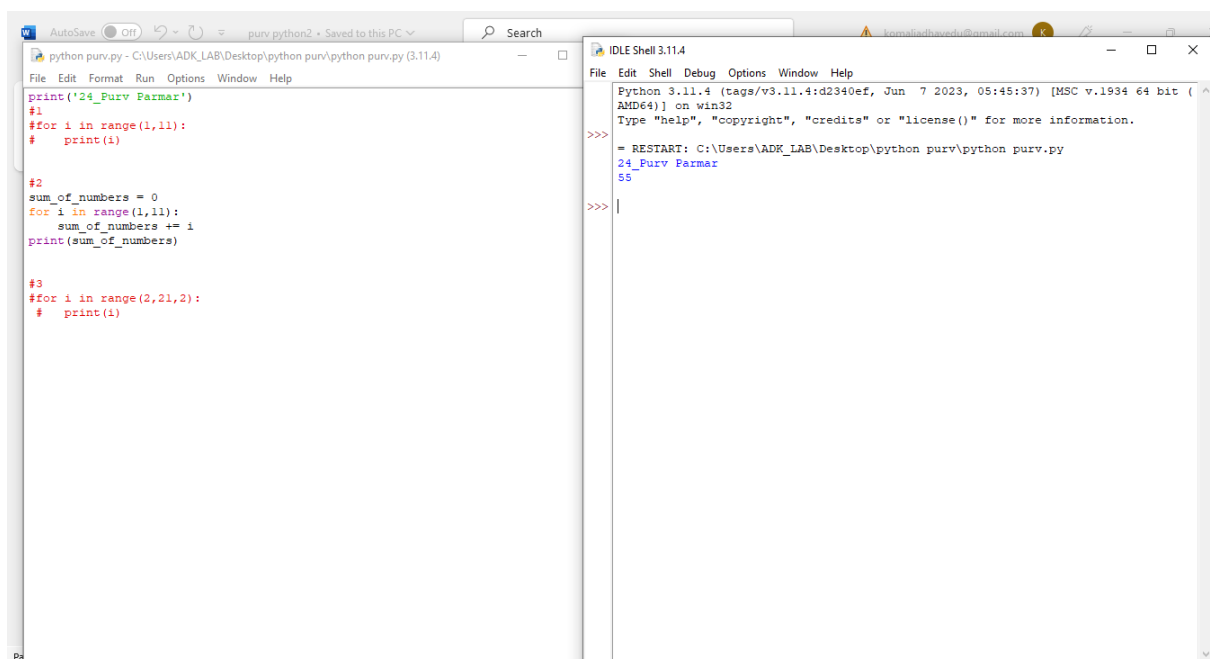
```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help
print('24_Purv Parmar')
#1
for i in range(1,11):
    print(i)

#2
sum_of_numbers = 0
for i in range(1,11):
    sum_of_numbers += i
print(sum_of_numbers)

#3
for i in range(2,21,2):
    print(i)
```

```
IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py =====
24_Purv Parmar
1
2
3
4
5
6
7
8
9
10
>>>
```

2. Sum of the first 10 natural numbers:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with three sections of code. The first section prints the name '24_Purv Parmar'. The second section uses a for loop to calculate the sum of the first 10 natural numbers. The third section uses a for loop to print numbers from 2 to 2. The right window shows the output of the script, which is the name '24_Purv Parmar' followed by the sum 55.

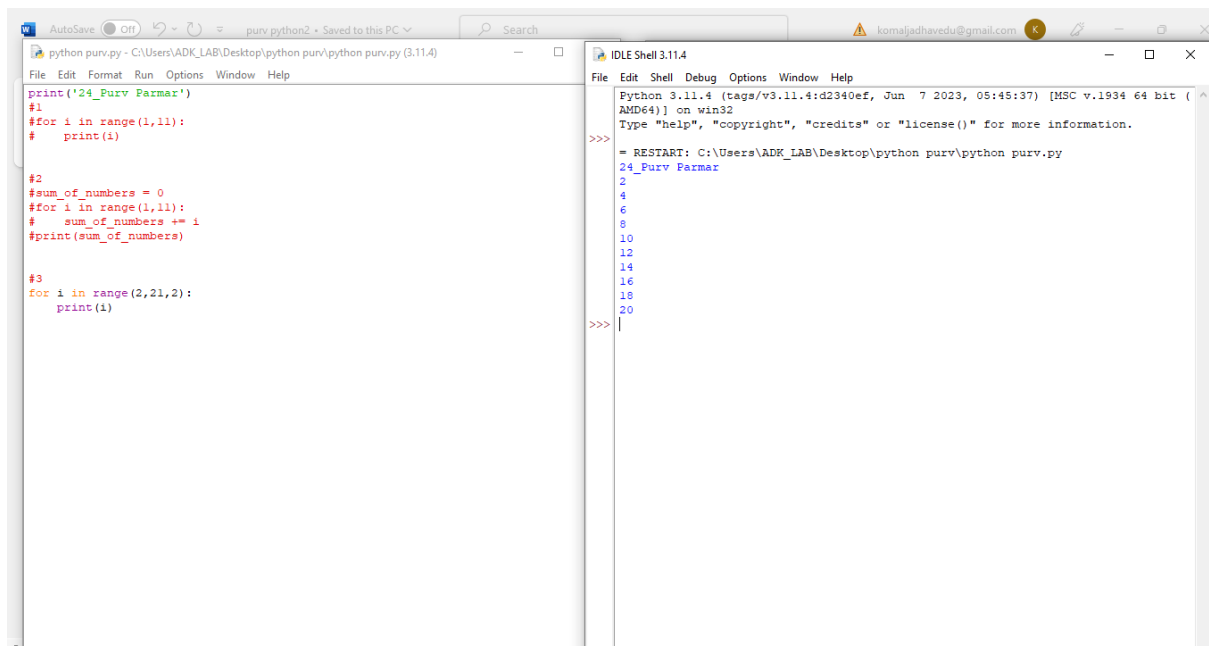
```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help
print('24_Purv Parmar')
#1
for i in range(1,11):
    print(i)

#2
sum_of_numbers = 0
for i in range(1,11):
    sum_of_numbers += i
print(sum_of_numbers)

#3
for i in range(2,21,2):
    print(i)
```

```
IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py =====
24_Purv Parmar
55
>>>
```

3. Printing the even numbers from 1 to 20:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with three sections: a header print, a loop to calculate the sum of numbers 1 to 11, and a loop to print even numbers from 2 to 20. The right window shows the execution output, which includes the header, the sum (66), and the even numbers (2, 4, 6, 8, 10, 12, 14, 16, 18, 20).

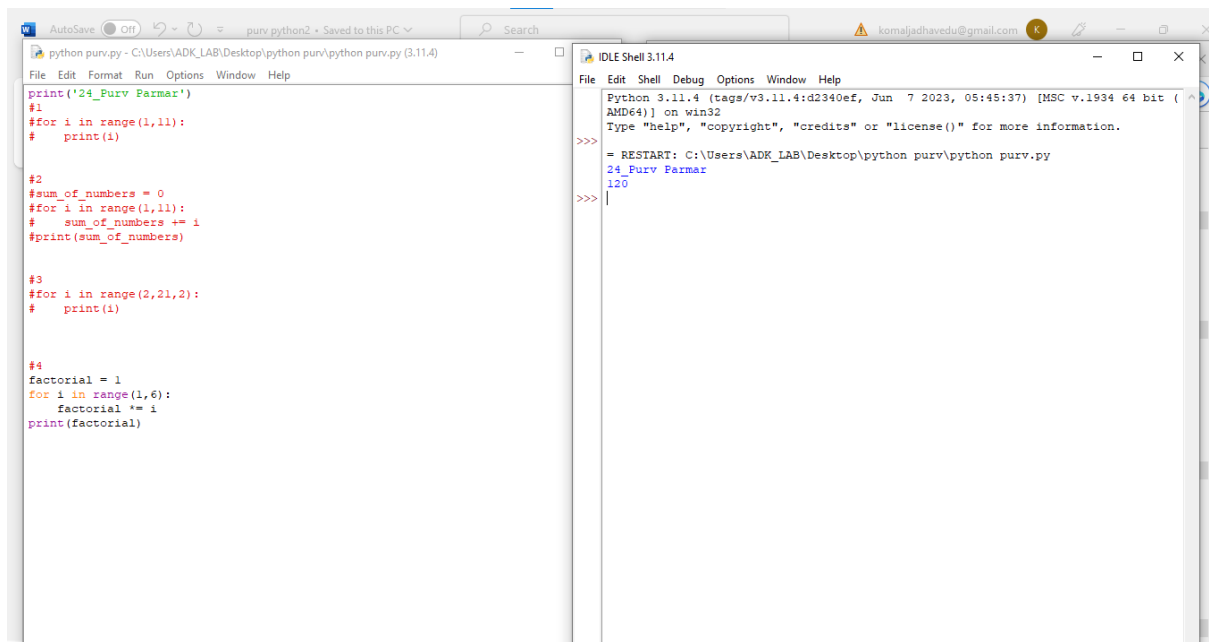
```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help
print('24_Purv Parmar')
#1
# for i in range(1,11):
#     print(i)

#2
#sum_of_numbers = 0
# for i in range(1,11):
#     sum_of_numbers += i
# print(sum_of_numbers)

#3
for i in range(2,21,2):
    print(i)
```

```
IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
- RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
2
4
6
8
10
12
14
16
18
20
>>>
```

4. Printing the factorial of 5:



The screenshot shows the same Python IDE with the script updated to calculate the factorial of 5. The left window shows the code for the factorial calculation. The right window shows the execution output, which includes the header, the sum (66), the even numbers (2, 4, 6, 8, 10, 12, 14, 16, 18, 20), and the factorial of 5 (120).

```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help
print('24_Purv Parmar')
#1
# for i in range(1,11):
#     print(i)

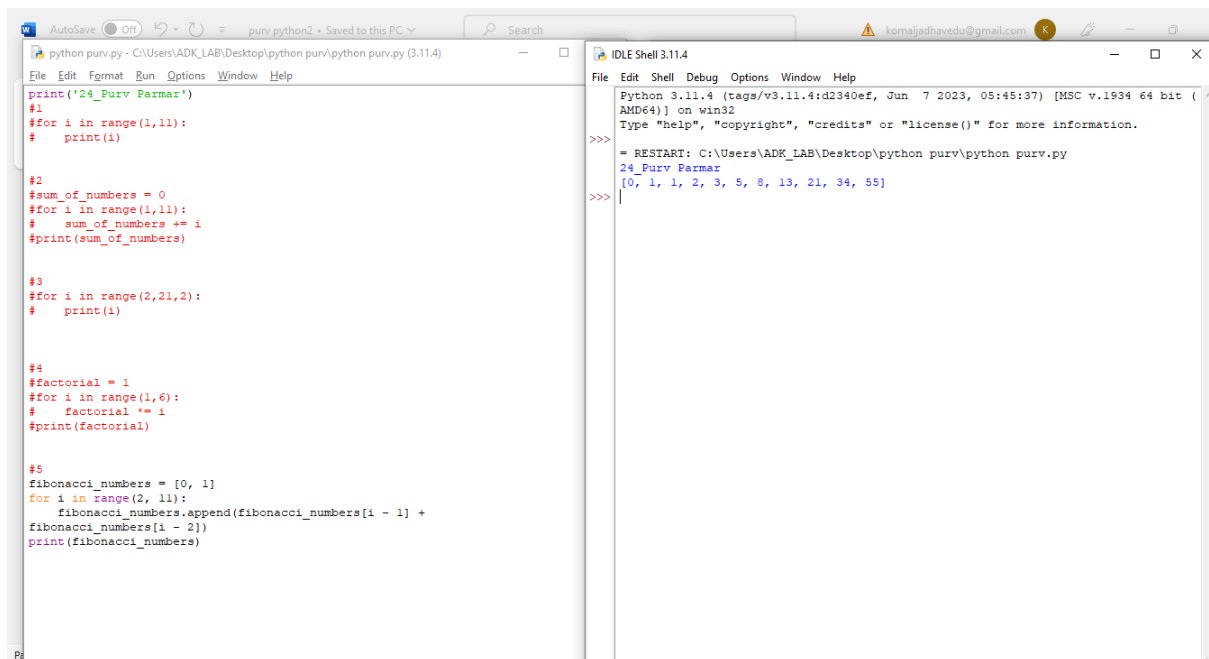
#2
#sum_of_numbers = 0
# for i in range(1,11):
#     sum_of_numbers += i
# print(sum_of_numbers)

#3
for i in range(2,21,2):
    print(i)

#4
factorial = 1
for i in range(1,6):
    factorial *= i
print(factorial)
```

```
IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
- RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
2
4
6
8
10
12
14
16
18
20
120
>>>
```

5. Printing the Fibonacci sequence up to the 10th term:



The screenshot shows a Python IDE with a file named 'python purv.py' and an IDLE Shell window. The code in the file defines a function '24_Purv Parmar' that prints the Fibonacci sequence up to the 10th term. The output in the shell window is '[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]'.

```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help

print('24_Purv Parmar')
#1
#for i in range(1,11):
#    print(i)

#2
#sum_of_numbers = 0
#for i in range(1,11):
#    sum_of_numbers += i
#print(sum_of_numbers)

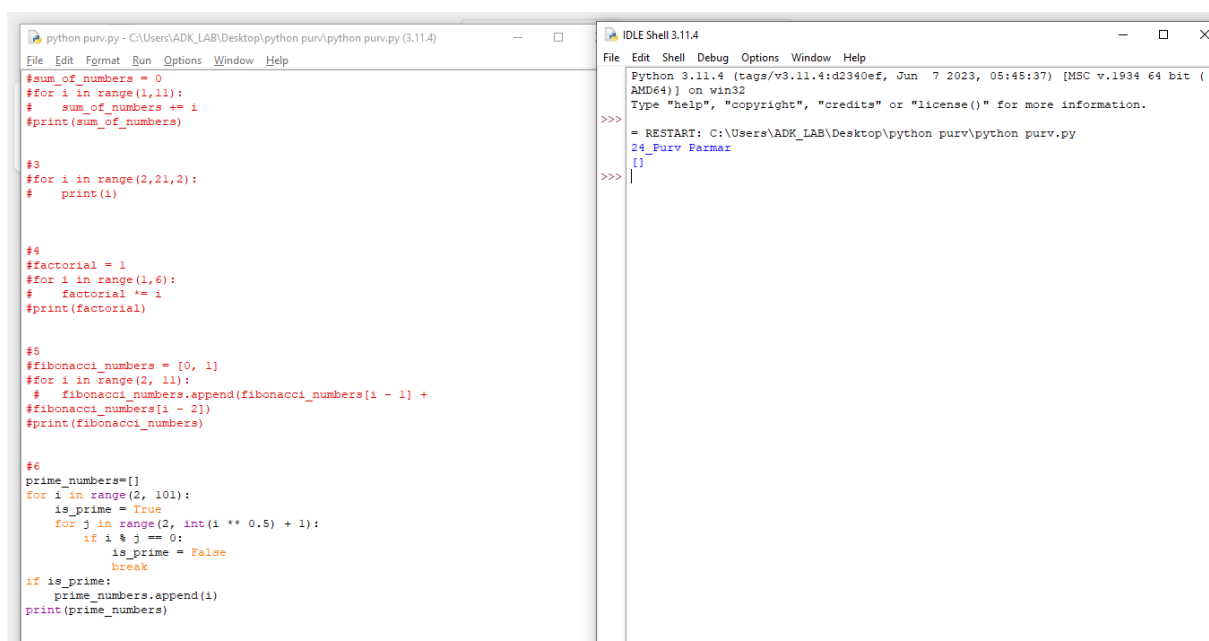
#3
#for i in range(2,21,2):
#    print(i)

#4
#factorial = 1
#for i in range(1,6):
#    factorial *= i
#print(factorial)

#5
fibonacci_numbers = [0, 1]
for i in range(2, 11):
    fibonacci_numbers.append(fibonacci_numbers[i - 1] +
                             fibonacci_numbers[i - 2])
print(fibonacci_numbers)

IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
>>>
```

6. Printing the prime numbers from 2 to 100:



The screenshot shows a Python IDE with a file named 'python purv.py' and an IDLE Shell window. The code in the file defines a function '24_Purv Parmar' that prints the prime numbers from 2 to 100. The output in the shell window is '[]'.

```
python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)
File Edit Format Run Options Window Help

#sum_of_numbers = 0
#for i in range(1,11):
#    sum_of_numbers += i
#print(sum_of_numbers)

#3
#for i in range(2,21,2):
#    print(i)

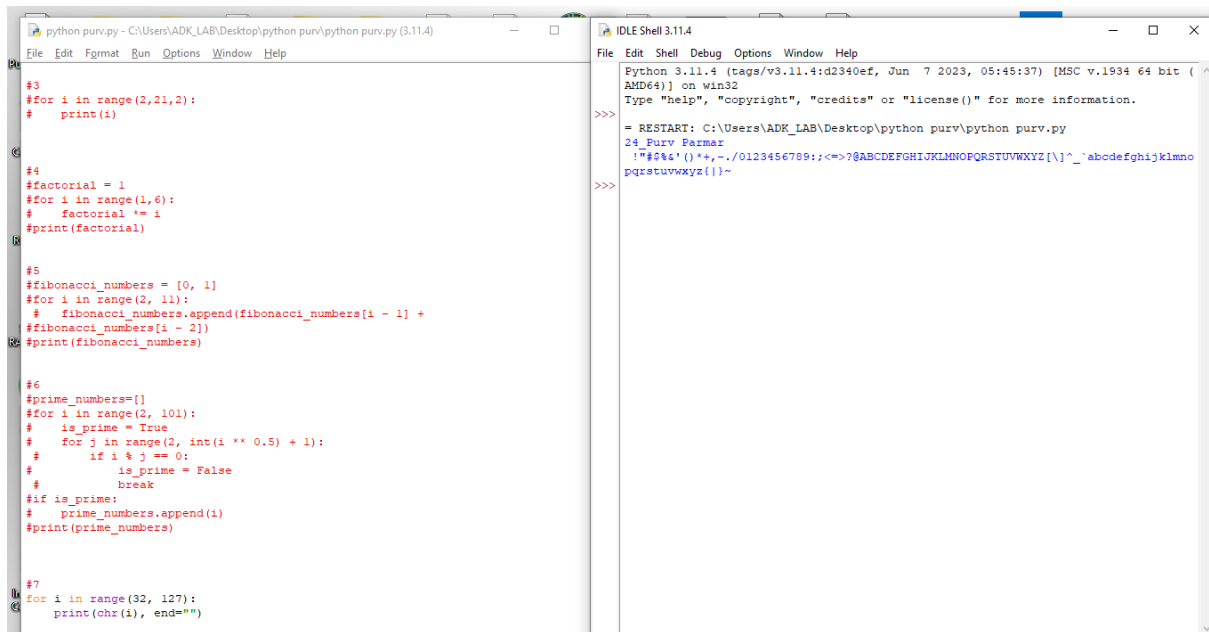
#4
#factorial = 1
#for i in range(1,6):
#    factorial *= i
#print(factorial)

#5
#fibonacci_numbers = [0, 1]
#for i in range(2, 11):
#    fibonacci_numbers.append(fibonacci_numbers[i - 1] +
#                             fibonacci_numbers[i - 2])
#print(fibonacci_numbers)

#6
prime_numbers=[]
for i in range(2, 101):
    is_prime = True
    for j in range(2, int(i ** 0.5) + 1):
        if i % j == 0:
            is_prime = False
            break
    if is_prime:
        prime_numbers.append(i)
print(prime_numbers)

IDLE Shell 3.11.4
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
[]
>>>
```

7. Printing the ASCII characters from 32 to 126:



The screenshot shows a Python IDE with two windows. The left window, titled 'python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)', contains the following code:

```
#3
# for i in range(2,21,2):
#     print(i)

#4
#factorial = 1
# for i in range(1,6):
#     factorial *= i
# print(factorial)

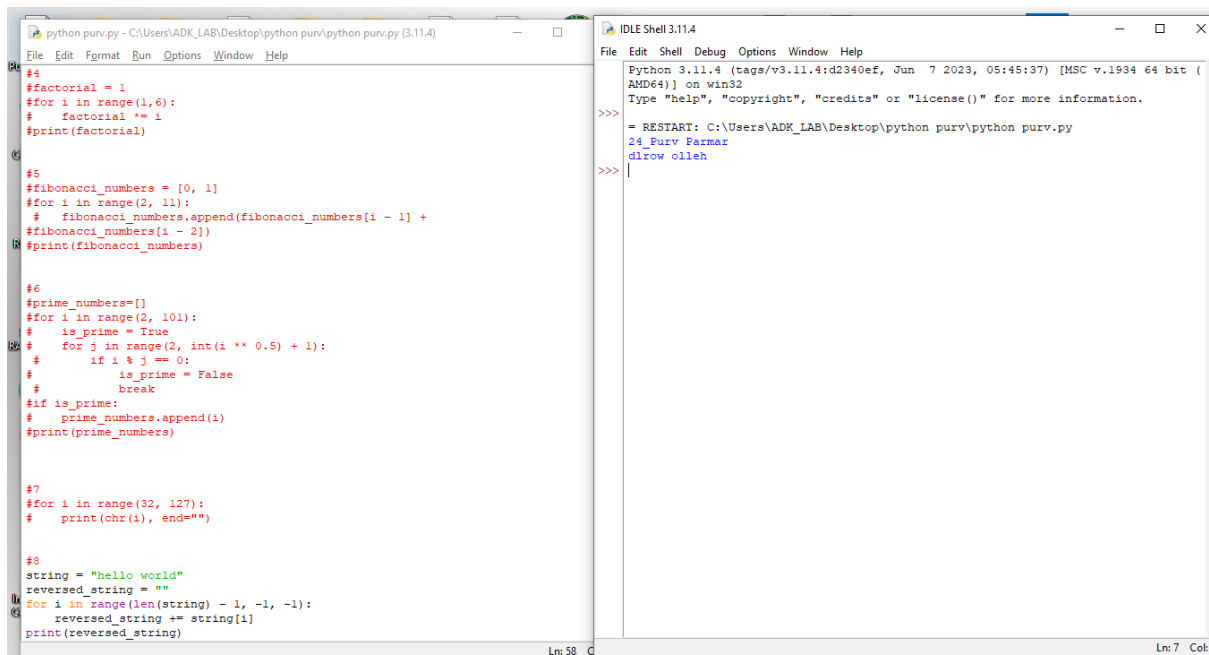
#5
#fibonacci_numbers = [0, 1]
# for i in range(2, 11):
#     fibonacci_numbers.append(fibonacci_numbers[i - 1] +
#                               fibonacci_numbers[i - 2])
# print(fibonacci_numbers)

#6
#prime_numbers=[]
# for i in range(2, 101):
#     is_prime = True
#     for j in range(2, int(i ** 0.5) + 1):
#         if i % j == 0:
#             is_prime = False
#             break
#     if is_prime:
#         prime_numbers.append(i)
# print(prime_numbers)

#7
for i in range(32, 127):
    print(chr(i), end="")
```

The right window, titled 'IDLE Shell 3.11.4', shows the output of the code, which is a string of all ASCII characters from 32 to 126, including spaces, punctuation, and alphanumeric characters.

8. Printing the reverse of a string:



The screenshot shows a Python IDE with two windows. The left window, titled 'python purv.py - C:\Users\ADK_LAB\Desktop\python purv\python purv.py (3.11.4)', contains the following code:

```
#4
#factorial = 1
# for i in range(1,6):
#     factorial *= i
# print(factorial)

#5
#fibonacci_numbers = [0, 1]
# for i in range(2, 11):
#     fibonacci_numbers.append(fibonacci_numbers[i - 1] +
#                               fibonacci_numbers[i - 2])
# print(fibonacci_numbers)

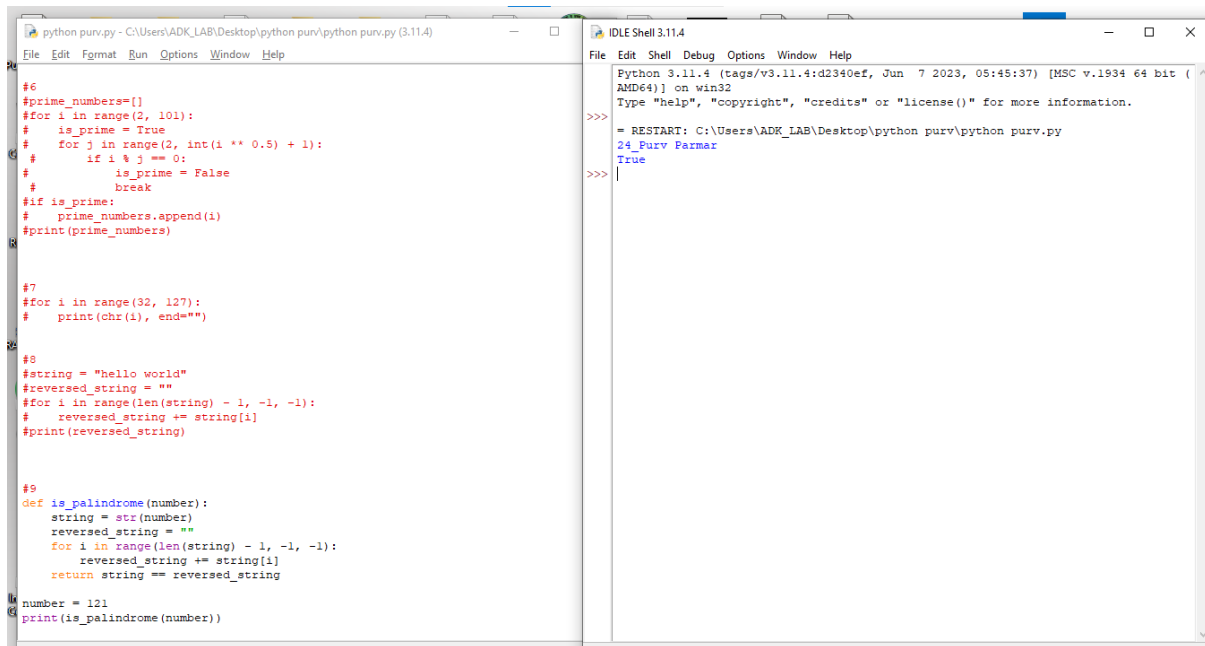
#6
#prime_numbers=[]
# for i in range(2, 101):
#     is_prime = True
#     for j in range(2, int(i ** 0.5) + 1):
#         if i % j == 0:
#             is_prime = False
#             break
#     if is_prime:
#         prime_numbers.append(i)
# print(prime_numbers)

#7
for i in range(32, 127):
    print(chr(i), end="")

#8
string = "hello world"
reversed_string = ""
for i in range(len(string) - 1, -1, -1):
    reversed_string += string[i]
print(reversed_string)
```

The right window, titled 'IDLE Shell 3.11.4', shows the output of the code, which is the string 'dlrow olleh'.

9. Checking if a number is a palindrome:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with several functions and a main execution block. The right window shows the IDLE Shell with the output of the script.

```
#6
#prime_numbers=[]
#for i in range(2, 101):
#    is_prime = True
#    for j in range(2, int(i ** 0.5) + 1):
#        if i % j == 0:
#            is_prime = False
#            break
#    if is_prime:
#        prime_numbers.append(i)
#print(prime_numbers)

#7
#for i in range(32, 127):
#    print(chr(i), end=" ")

#8
#string = "hello world"
#reversed_string = ""
#for i in range(len(string) - 1, -1, -1):
#    reversed_string += string[i]
#print(reversed_string)

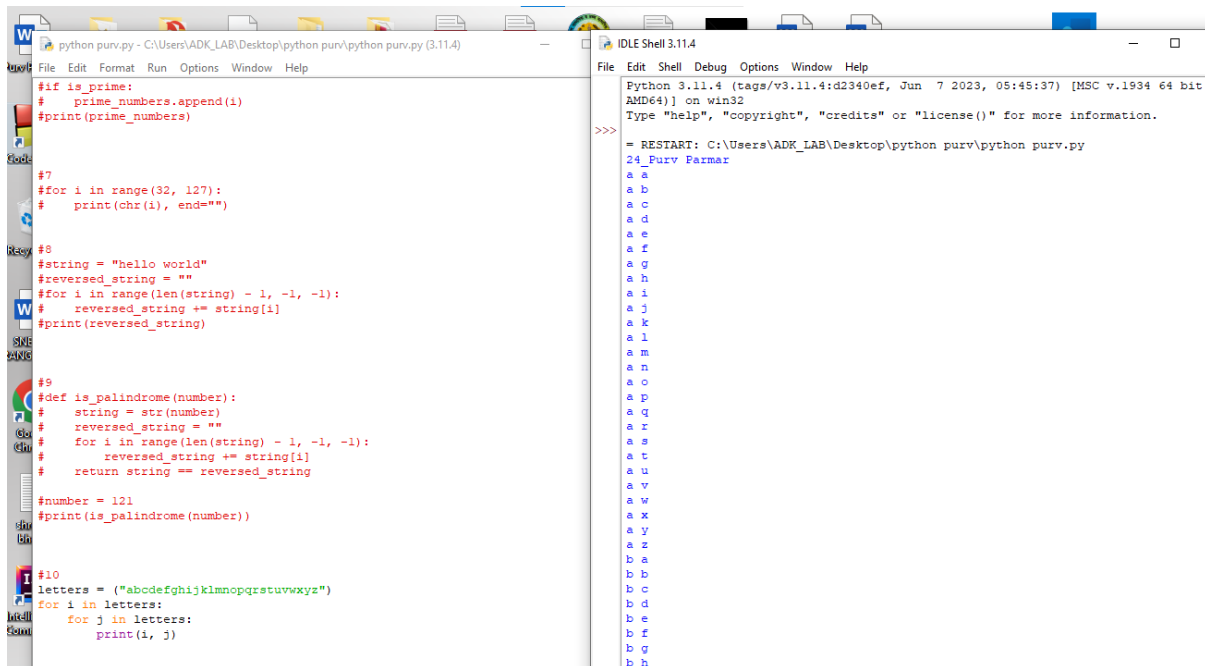
#9
def is_palindrome(number):
    string = str(number)
    reversed_string = ""
    for i in range(len(string) - 1, -1, -1):
        reversed_string += string[i]
    return string == reversed_string

number = 121
print(is_palindrome(number))
```

The IDLE Shell output shows the following:

```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
True
>>>
```

10. Printing all the possible combinations of two letters:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with several functions and a main execution block. The right window shows the IDLE Shell with the output of the script.

```
#if is_prime:
#    prime_numbers.append(i)
#print(prime_numbers)

#7
#for i in range(32, 127):
#    print(chr(i), end=" ")

#8
#string = "hello world"
#reversed_string = ""
#for i in range(len(string) - 1, -1, -1):
#    reversed_string += string[i]
#print(reversed_string)

#9
def is_palindrome(number):
    string = str(number)
    reversed_string = ""
    for i in range(len(string) - 1, -1, -1):
        reversed_string += string[i]
    return string == reversed_string

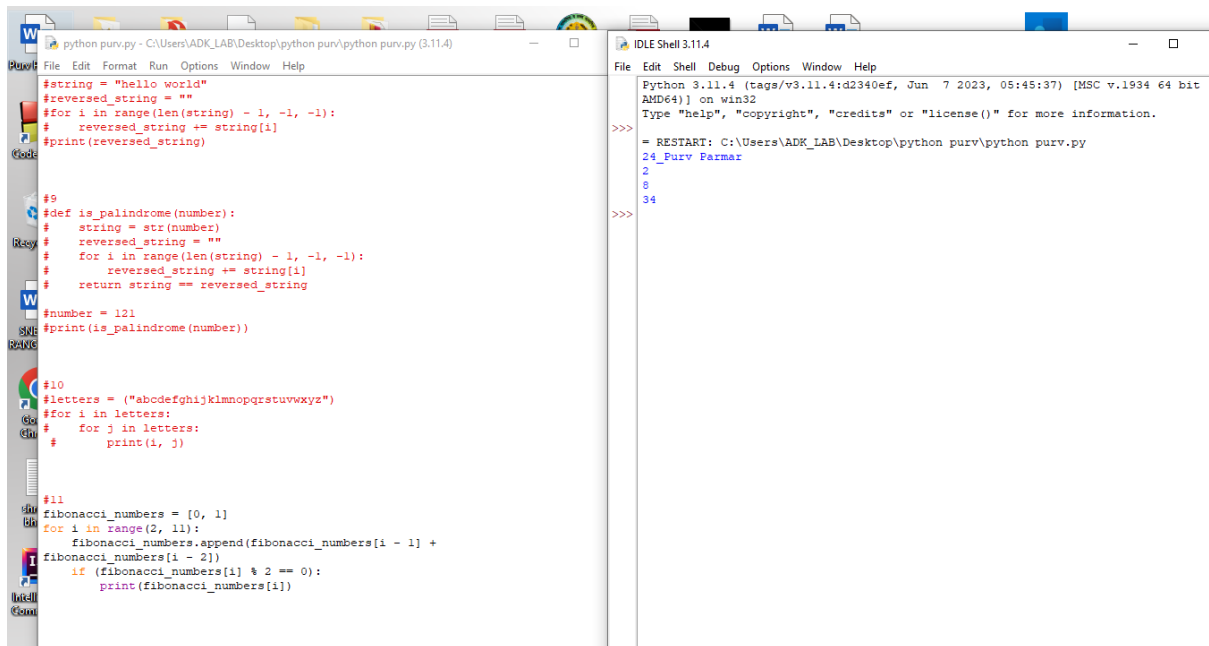
number = 121
print(is_palindrome(number))

#10
letters = ("abcdefghijklmnopqrstuvwxyz")
for i in letters:
    for j in letters:
        print(i, j)
```

The IDLE Shell output shows the following:

```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
a a
a b
a c
a d
a e
a f
a g
a h
a i
a j
a k
a l
a m
a n
a o
a p
a q
a r
a s
a t
a u
a v
a w
a x
a y
a z
b a
b b
b c
b d
b e
b f
b g
b h
```

11. Printing the first 10 even Fibonacci numbers:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with several functions and a loop to print the first 10 even Fibonacci numbers. The right window shows the output of the script, which is the first 10 even Fibonacci numbers: 2, 8, 34, 144, 610, 2584, 10946, 46368, 196410, and 832040.

```
#string = "hello world"
#reversed_string = ""
# for i in range(len(string) - 1, -1, -1):
#     reversed_string += string[i]
#print(reversed_string)

#9
#def is_palindrome(number):
#     string = str(number)
#     reversed_string = ""
#     for i in range(len(string) - 1, -1, -1):
#         reversed_string += string[i]
#     return string == reversed_string

#number = 121
#print(is_palindrome(number))

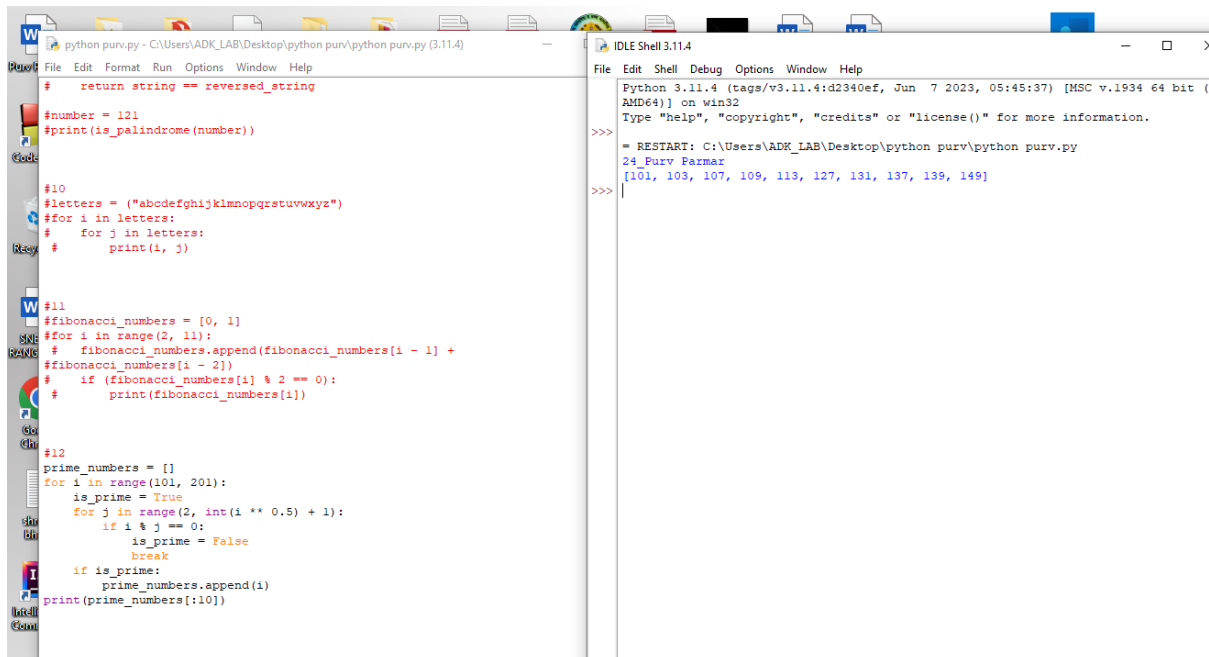
#10
#letters = ("abcdefghijklmnopqrstuvwxyz")
# for i in letters:
#     for j in letters:
#         print(i, j)

#11
fibonacci_numbers = [0, 1]
for i in range(2, 11):
    fibonacci_numbers.append(fibonacci_numbers[i - 1] +
                             fibonacci_numbers[i - 2])
if (fibonacci_numbers[i] % 2 == 0):
    print(fibonacci_numbers[i])
```

```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit
AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> = RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
2
8
34
>>>
```

12. Finding the first 10 prime numbers greater than 100:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with a function to check if a number is prime and a loop to find the first 10 prime numbers greater than 100. The right window shows the output of the script, which is the first 10 prime numbers greater than 100: 101, 103, 107, 109, 113, 127, 131, 137, 139, and 149.

```
#     return string == reversed_string

#number = 121
#print(is_palindrome(number))

#10
#letters = ("abcdefghijklmnopqrstuvwxyz")
# for i in letters:
#     for j in letters:
#         print(i, j)

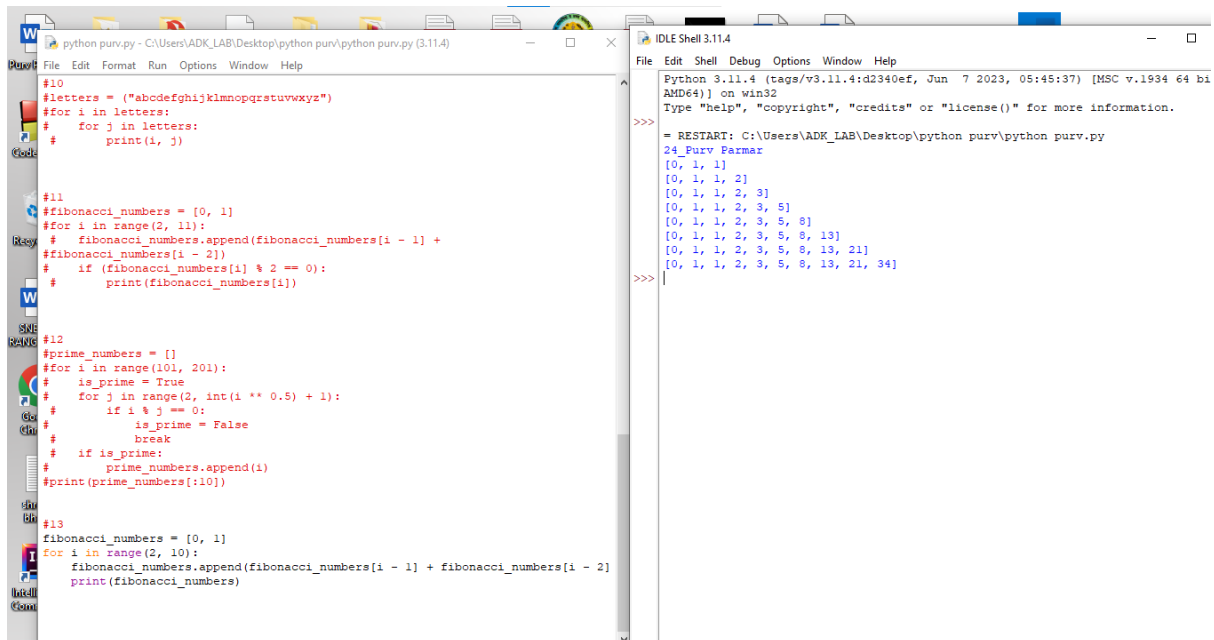
#11
#fibonacci_numbers = [0, 1]
# for i in range(2, 11):
#     # fibonacci_numbers.append(fibonacci_numbers[i - 1] +
#     # fibonacci_numbers[i - 2])
#     # if (fibonacci_numbers[i] % 2 == 0):
#     #     print(fibonacci_numbers[i])

#12
prime_numbers = []
for i in range(101, 201):
    is_prime = True
    for j in range(2, int(i ** 0.5) + 1):
        if i % j == 0:
            is_prime = False
            break
    if is_prime:
        prime_numbers.append(i)
print(prime_numbers[:10])
```

```
Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit
AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>> = RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
[101, 103, 107, 109, 113, 127, 131, 137, 139, 149]
>>>
```

13.Printing the Fibonacci sequence in a spiral:



The screenshot shows a Python IDE with two windows. The left window displays a Python script with three sections: printing letters, generating Fibonacci numbers, and generating prime numbers. The right window shows the output of the script, which includes the Fibonacci sequence in a spiral format.

```
#10
#letters = ("abcdefghijklmnopqrstuvwxyz")
#for i in letters:
#    for j in letters:
#        print(i, j)

#11
#fibonacci_numbers = [0, 1]
#for i in range(2, 11):
#    fibonacci_numbers.append(fibonacci_numbers[i - 1] +
#                             fibonacci_numbers[i - 2])
#    if (fibonacci_numbers[i] % 2 == 0):
#        print(fibonacci_numbers[i])

#12
#prime_numbers = []
#for i in range(101, 201):
#    is_prime = True
#    for j in range(2, int(i ** 0.5) + 1):
#        if i % j == 0:
#            is_prime = False
#            break
#    if is_prime:
#        prime_numbers.append(i)
#print(prime_numbers[:10])

#13
fibonacci_numbers = [0, 1]
for i in range(2, 10):
    fibonacci_numbers.append(fibonacci_numbers[i - 1] + fibonacci_numbers[i - 2])
    print(fibonacci_numbers)
```

Python 3.11.4 (tags/v3.11.4:d2340ef, Jun 7 2023, 05:45:37) [MSC v.1934 64 bit AMD64] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADK_LAB\Desktop\python purv\python purv.py
24_Purv Parmar
[0, 1, 1]
[0, 1, 1, 2]
[0, 1, 1, 2, 3]
[0, 1, 1, 2, 3, 5]
[0, 1, 1, 2, 3, 5, 8]
[0, 1, 1, 2, 3, 5, 8, 13]
[0, 1, 1, 2, 3, 5, 8, 13, 21]
[0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
>>>