Date: February 14, 2024

SMART INTERNZ - APSCHE

AI / ML Training

Assessment - 1

NAME - CHALLA GOPICHAND

EMAIL ID - gopichand0816@gmail.com

PHONE NO - 9391895115

ROLL NO - 20A41A0516

COLLEGE - LOYOLA INSTITUTE OF TECHNOLOGY AND MANAGEMENT

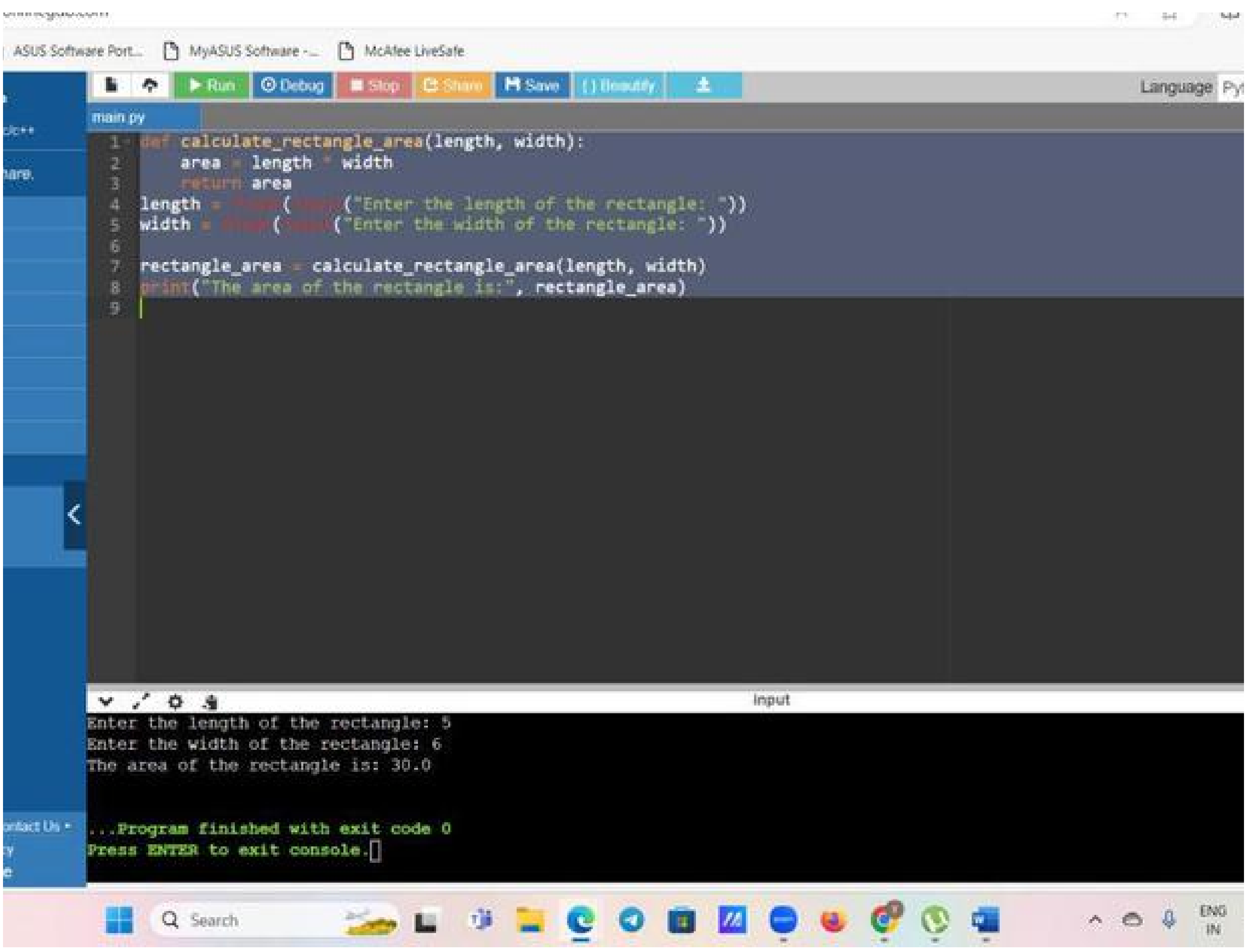
---

1. Write a Python program to calculate the area of a rectangle given its length and width.

Source code:

```
def calculate_rectangle_area(length, width):



area = length * width



return area



length = float(input("Enter the length of the rectangle: "))



width = float(input("Enter the width of the rectangle: "))





rectangle_area = calculate_rectangle_area(length, width)



print("The area of the rectangle is:", rectangle_area)
```

Output:



```
def calculate_rectangle_area(length, width):
    area = length * width
    return area
length = float(input("Enter the length of the rectangle: "))
width = float(input("Enter the width of the rectangle: "))

rectangle_area = calculate_rectangle_area(length, width)
print("The area of the rectangle is:", rectangle_area)
```

```
Enter the length of the rectangle: 5
Enter the width of the rectangle: 6
The area of the rectangle is: 30.0


...Program finished with exit code 0
Press ENTER to exit console.
```

2.Write a program to convert miles to kilometers.

<u>Source code:</u>

def miles_to_kilometers(miles):

kilometers = miles * 1.60934

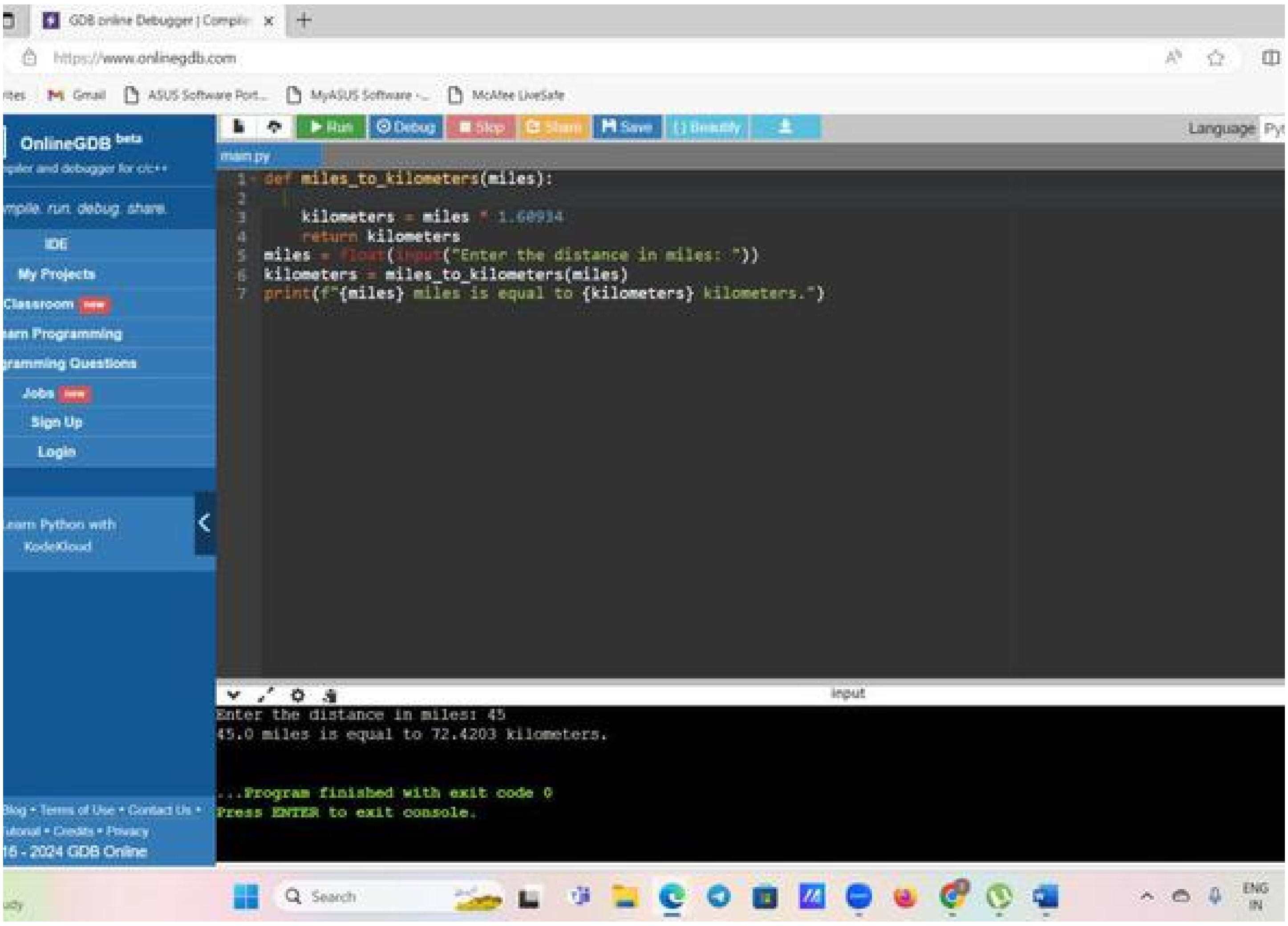
return kilometers

miles = float(input("Enter the distance in miles: "))

kilometers = miles_to_kilometers(miles)

print(f"{miles} miles is equal to {kilometers} kilometers.")

Output:



```
def miles_to_kilometers(miles):

    kilometers = miles * 1.60934
    return kilometers
miles = float(input("Enter the distance in miles: "))
kilometers = miles_to_kilometers(miles)
print(f"{miles} miles is equal to {kilometers} kilometers.")
```

```
Enter the distance in miles: 45
45.0 miles is equal to 72.4203 kilometers.


...Program finished with exit code 0
Press ENTER to exit console.
```

3.Write a function to check if a given string is a palindrome.

Source code:

def is_palindrome(s):

s = s.replace(" ", "").lower()

return s == s[::-1]

string = input("Enter a string: ")

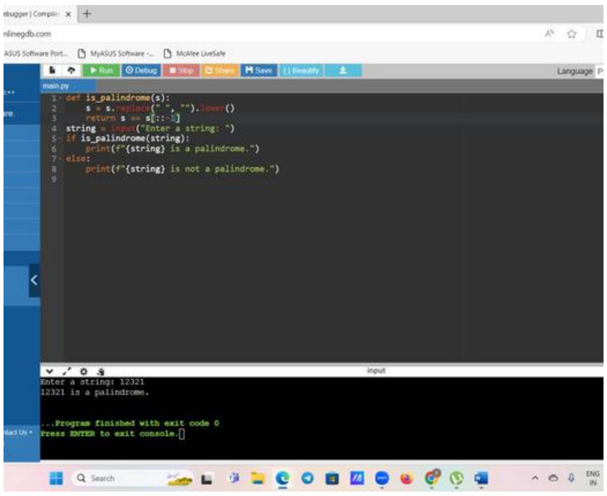
if is_palindrome(string):

print(f"{string} is a palindrome.")

else:

print(f"{string} is not a palindrome.")

<u>Output:</u>



```
def is_palindrome(s):
    s = s.replace(" ", "").lower()
    return s == s[::-1]
string = input("Enter a string: ")
if is_palindrome(string):
    print(f"{string} is a palindrome.")
else:
    print(f"{string} is not a palindrome.")
```

```
Enter a string: 12321
12321 is a palindrome.


...Program finished with exit code 0
Press ENTER to exit console.
```

4. Write a Python program to find the second largest element in a list.

<u>Source code:</u>

def find_second_largest(nums):

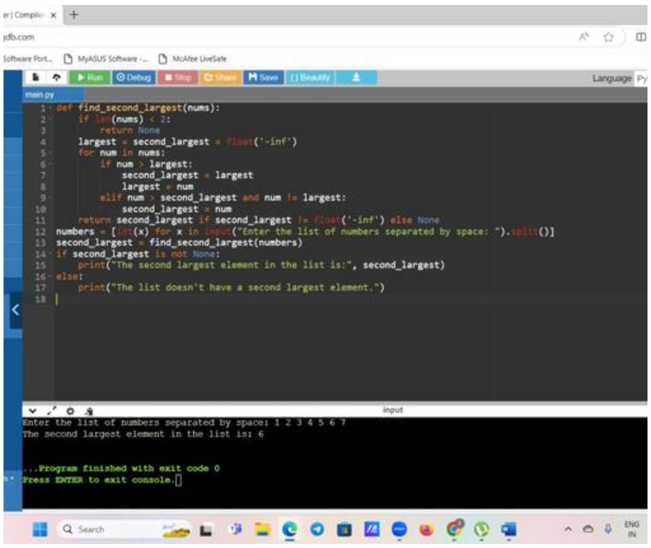
if len(nums) < 2:

return None

largest = second_largest = float('-inf')

```
for num in nums:

if num > largest:

second_largest = largest

largest = num

elif num > second_largest and num != largest:

second_largest = num

return second_largest if second_largest != float('-inf') else None

numbers = [int(x) for x in input("Enter the list of numbers separated by space: ").split()]

second_largest = find_second_largest(numbers)

if second_largest is not None:

print("The second largest element in the list is:", second_largest)

else:

print("The list doesn't have a second largest element.")
```

Output:



```python
def find_second_largest(nums):
    if len(nums) < 2:
        return None
    largest = second_largest = float('-inf')
    for num in nums:
        if num > largest:
            second_largest = largest
            largest = num
        elif num > second_largest and num != largest:
            second_largest = num
    return second_largest if second_largest != float('-inf') else None
numbers = [int(x) for x in input("Enter the list of numbers separated by space: ").split()]
second_largest = find_second_largest(numbers)
if second_largest is not None:
    print("The second largest element in the list is:", second_largest)
else:
    print("The list doesn't have a second largest element.")
```

```
Enter the list of numbers separated by space: 1 2 3 4 5 6 7
The second largest element in the list is: 6


...Program finished with exit code 0
Press ENTER to exit console.
```

5. Explain what indentation means in Python.

Indentation is a very important concept of Python because without properly indenting the Python code, you will end up seeing Indentation Error and the code will not get compiled.

Python indentation refers to adding white space before a statement to a particular block of code. In another word, all the statements with the same space to the right, belong to the same code block.

Python indentation is a way of telling a Python interpreter that the group of statements belongs to a particular block of code. A block is a combination of all these statements. Block can be regarded as the grouping of statements for a specific purpose. Most programming languages like C, C++, and Java use braces { } to define a block of code. Python uses indentation to highlight the blocks of code. Whitespace is used for indentation in Python. All statements with the same distance to the right belong to the same block of code. If a block has to be more deeply nested, it is simply indented further to the right.

For ex :- Statement (line 1), if condition (line 2), and statement (last line) belongs to the same block which means that after statement 1, if condition will be executed. and suppose the if condition becomes False then the Python will jump to the last statement for execution.

6.Write a program to perform set difference operation.

Source code:

def set_difference(set1, set2):

return set1 - set2

set1 = {1, 2, 3, 4, 5}

set2 = {4, 5, 6, 7, 8}

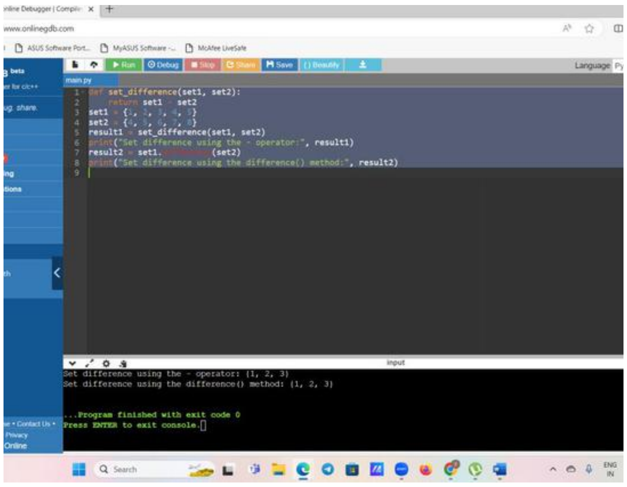
result1 = set_difference(set1, set2)

print("Set difference using the - operator:", result1)

result2 = set1.difference(set2)

print("Set difference using the difference() method:", result2)

<u>Output:</u>



```
def set_difference(set1, set2):
    return set1 - set2
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}
result1 = set_difference(set1, set2)
print("Set difference using the - operator:", result1)
result2 = set1.difference(set2)
print("Set difference using the difference() method:", result2)
```

```
Set difference using the - operator: {1, 2, 3}
Set difference using the difference() method: {1, 2, 3}


...Program finished with exit code 0
Press ENTER to exit console.
```

7.Write a Python program to print numbers from 1 to 10 using a while loop.

<u>Source code:</u>

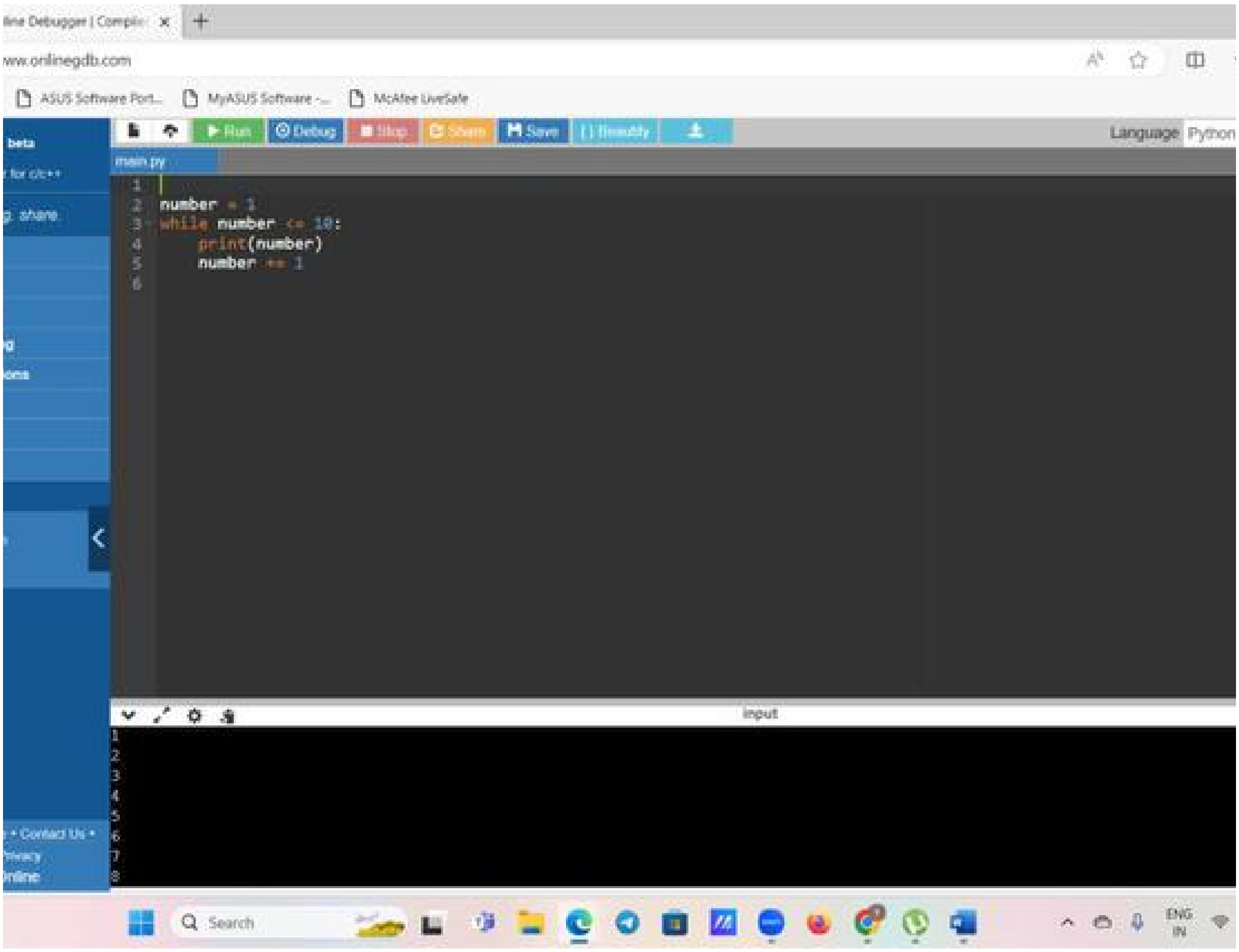
number = 1

while number <= 10:

print(number)

number += 1

<u>Output:</u>



```
number = 1
while number <= 10:
    print(number)
    number += 1
```

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

8.Write a program to calculate the factorial of a number using a while loop.

Source code:

def factorial(n):

if n < 0:

return None

elif n == 0:

return 1

else:

result = 1

while n > 0:

result *= n

a. **-= 1 return result**

number = int(input("Enter a number to calculate its factorial: "))

fact = factorial(number)

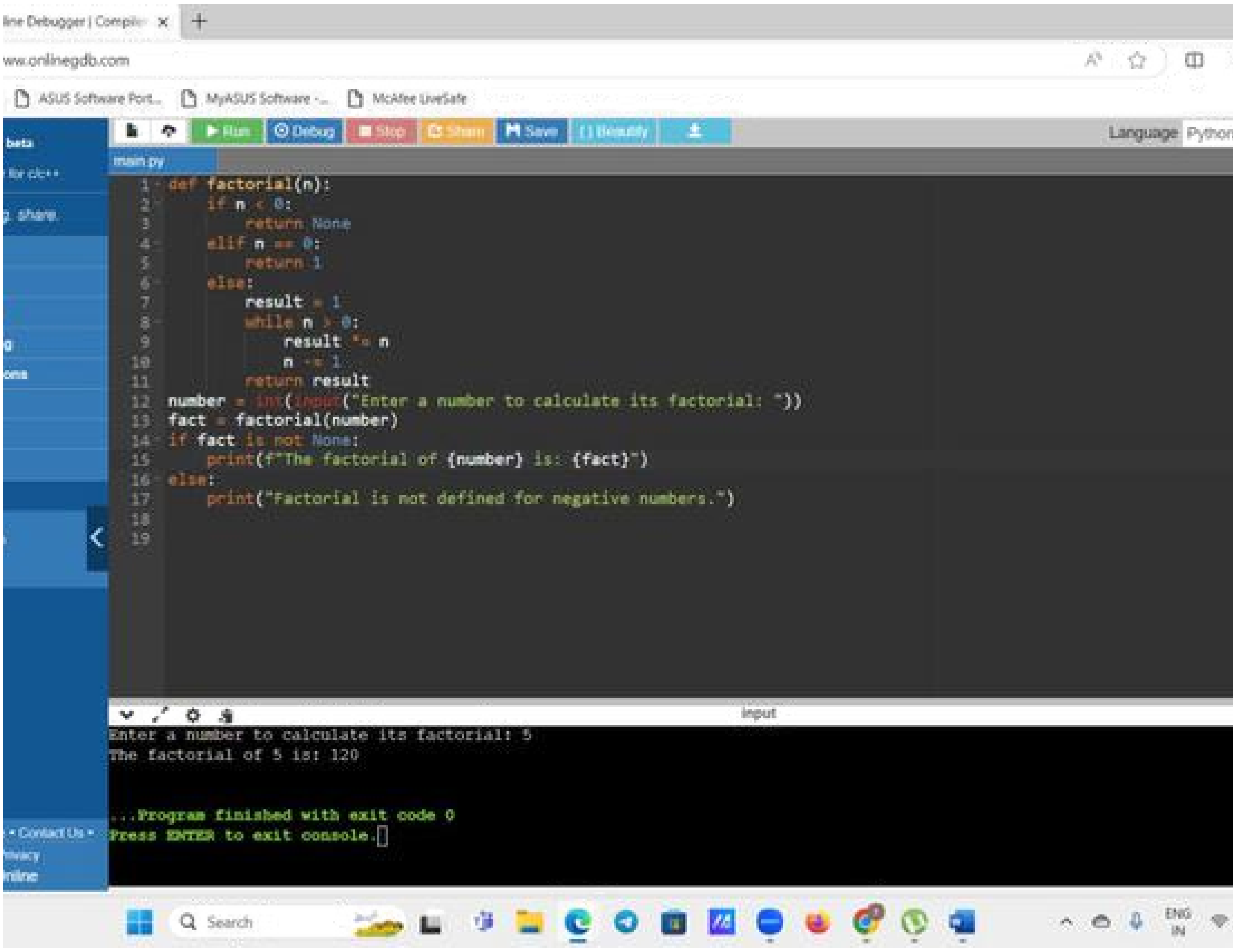
if fact is not None:

print(f"The factorial of {number} is: {fact}")

else:

print("Factorial is not defined for negative numbers.")

<u>Output:</u>



```python
def factorial(n):
    if n < 0:
        return None
    elif n == 0:
        return 1
    else:
        result = 1
        while n > 0:
            result *= n
            n -= 1
        return result
number = int(input("Enter a number to calculate its factorial: "))
fact = factorial(number)
if fact is not None:
    print(f"The factorial of {number} is: {fact}")
else:
    print("Factorial is not defined for negative numbers.")
```

```
Enter a number to calculate its factorial: 5
The factorial of 5 is: 120


...Program finished with exit code 0
Press ENTER to exit console.
```

9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else statements.

Source code:

def check_number(num):

if num > 0:

return "Positive"

elif num < 0:

return "Negative"

else:

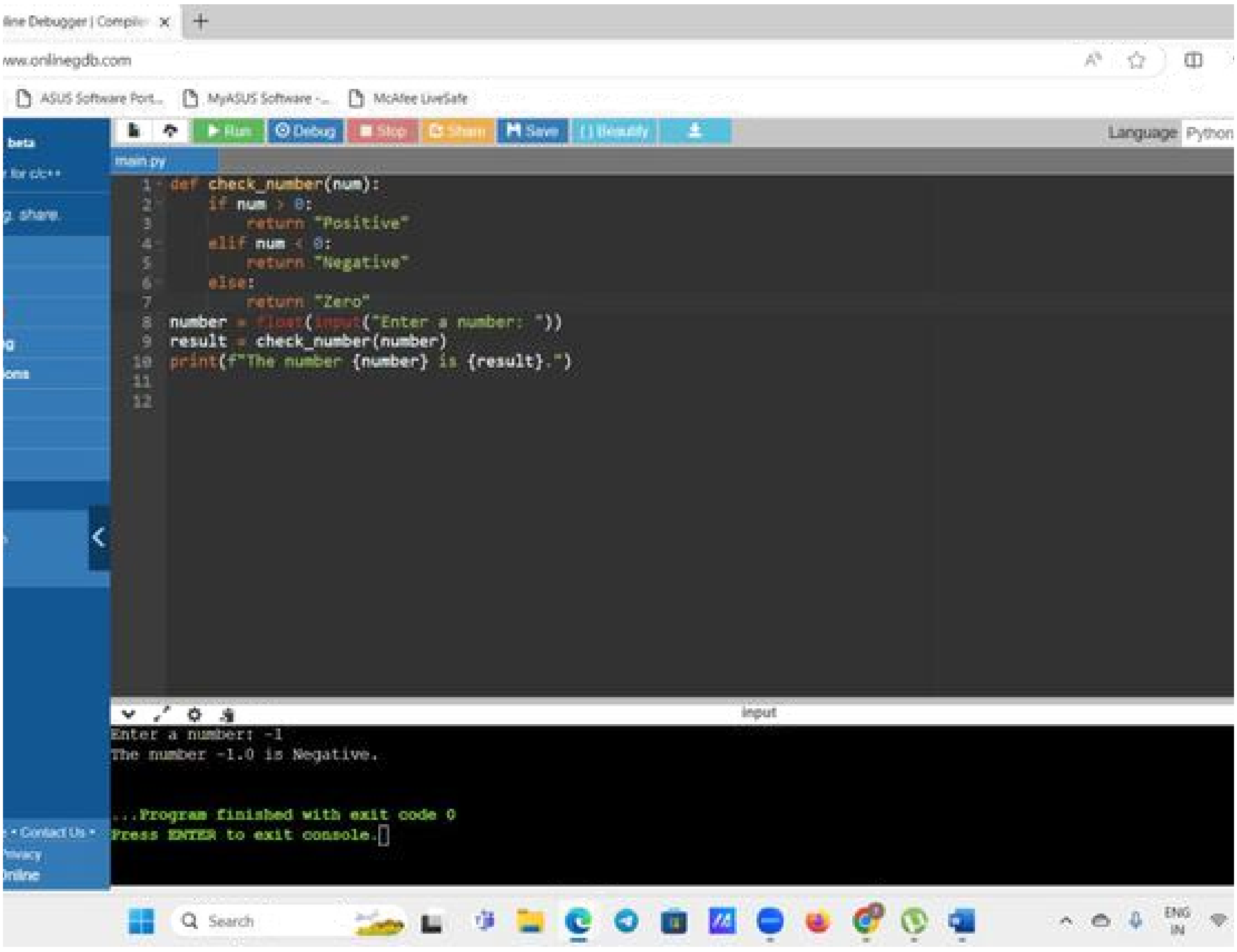
return "Zero"

number = float(input("Enter a number: "))

result = check_number(number)

print(f"The number {number} is {result}.")

<u>Output:</u>



```
def check_number(num):
    if num > 0:
        return "Positive"
    elif num < 0:
        return "Negative"
    else:
        return "Zero"
number = float(input("Enter a number: "))
result = check_number(number)
print(f"The number {number} is {result}.")
```

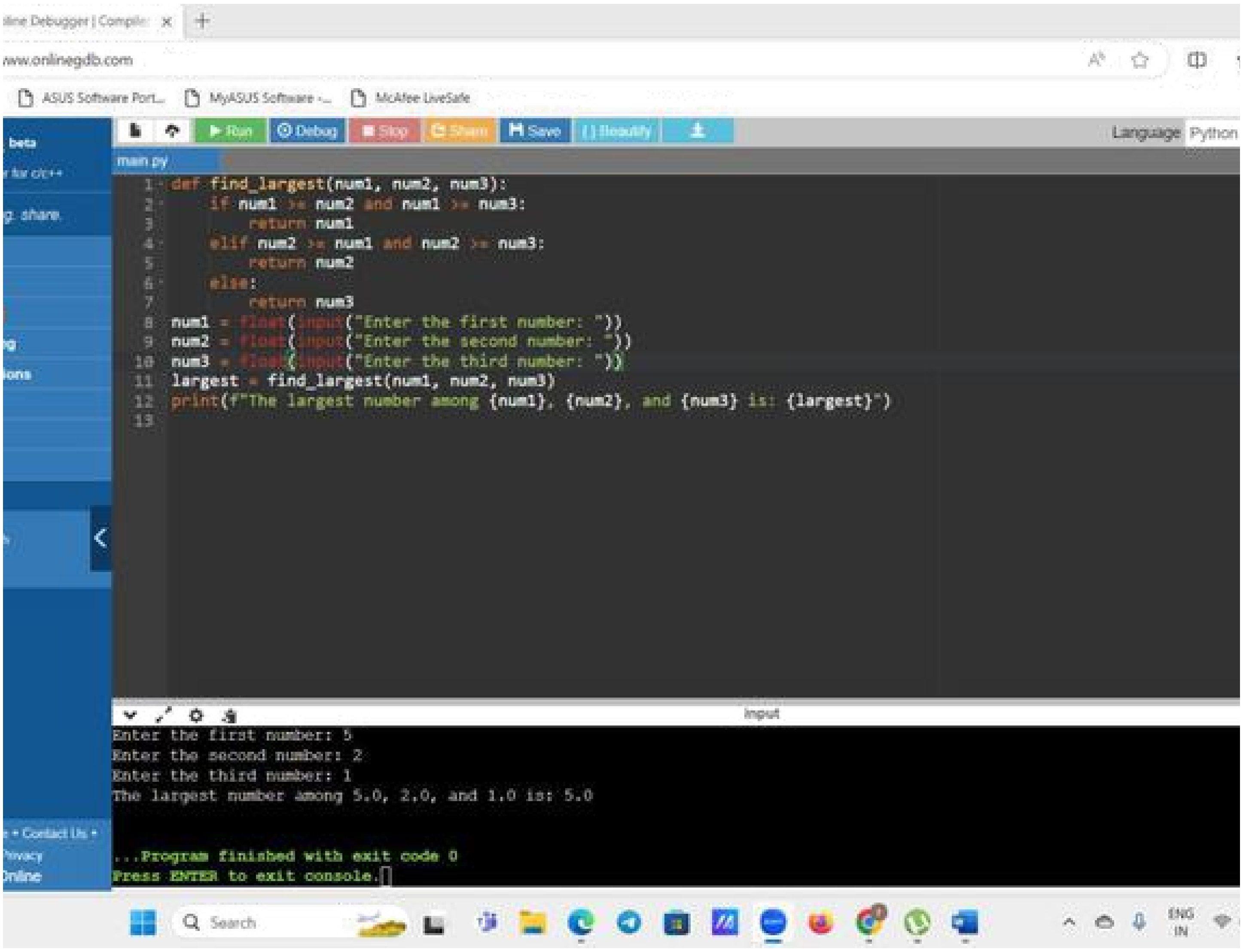
```
Enter a number: -1
The number -1.0 is Negative.


...Program finished with exit code 0
Press ENTER to exit console.
```

10.Write a program to determine the largest among three numbers using conditional statements.

<u>Source code:</u>

```
def find_largest(num1, num2, num3):

if num1 >= num2 and num1 >= num3:

return num1

elif num2 >= num1 and num2 >= num3:

return num2

else:

return num3

num1 = float(input("Enter the first number: "))

num2 = float(input("Enter the second number: "))

num3 = float(input("Enter the third number: "))

largest = find_largest(num1, num2, num3)

print(f"The largest number among {num1}, {num2}, and {num3} is: {largest}")
```

<u>Output:</u>



```
def find_largest(num1, num2, num3):
    if num1 >= num2 and num1 >= num3:
        return num1
    elif num2 >= num1 and num2 >= num3:
        return num2
    else:
        return num3
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))
largest = find_largest(num1, num2, num3)
print(f"The largest number among {num1}, {num2}, and {num3} is: {largest}")
```

```
Enter the first number: 5
Enter the second number: 2
Enter the third number: 1
The largest number among 5.0, 2.0, and 1.0 is: 5.0


...Program finished with exit code 0
Press ENTER to exit console.
```

11.Write a Python program to create a numpy array filled with ones of given shape.

Source code:

import numpy as np

def ones_array(shape):

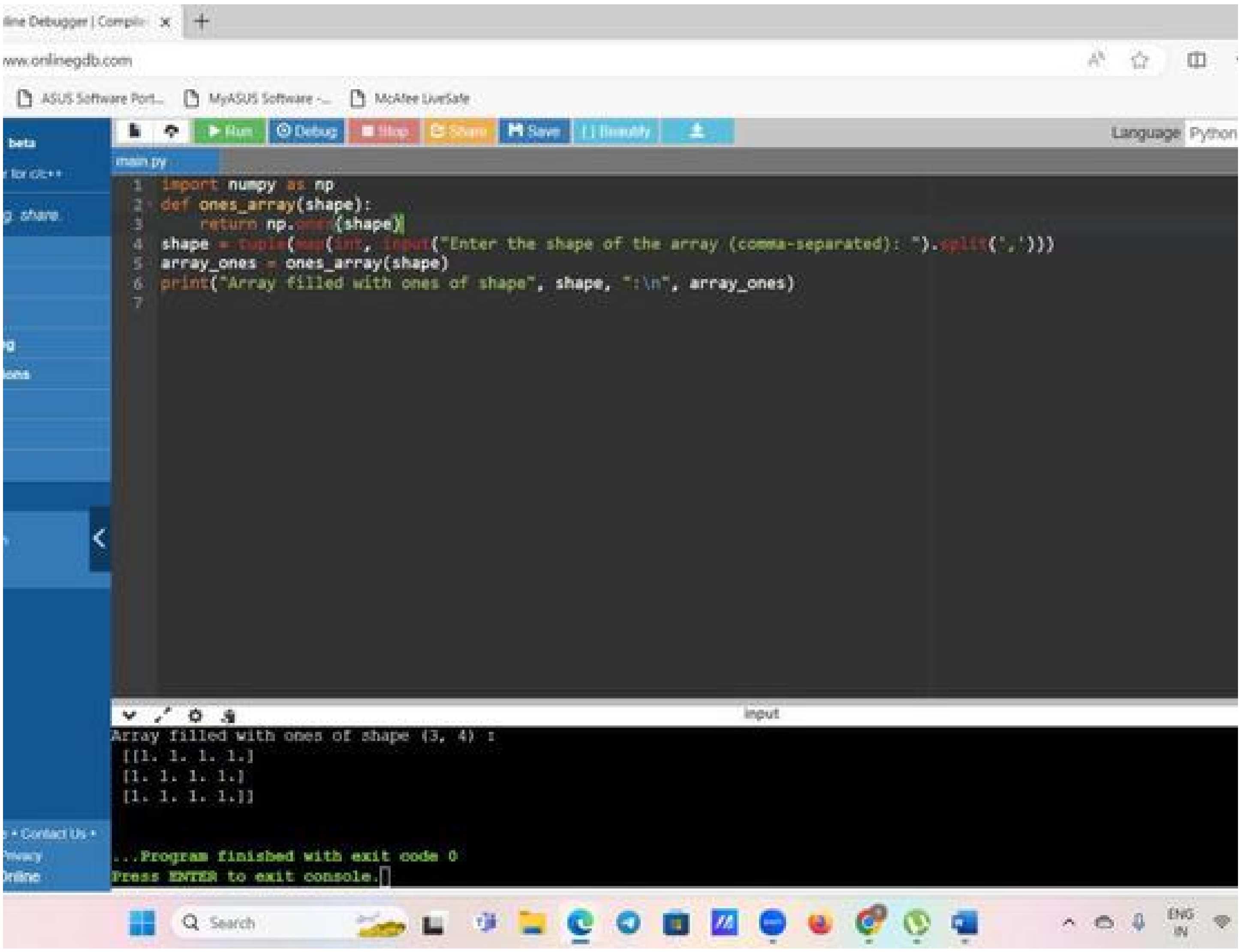
return np.ones(shape)

shape = tuple(map(int, input("Enter the shape of the array (comma-separated): ").split(',')))

array_ones = ones_array(shape)

print("Array filled with ones of shape", shape, ":\n", array_ones)

<u>Output:</u>



```
import numpy as np
def ones_array(shape):
    return np.ones(shape)
shape = tuple(map(int, input("Enter the shape of the array (comma-separated): ").split(',')))
array_ones = ones_array(shape)
print("Array filled with ones of shape", shape, ":\n", array_ones)
```

```
Array filled with ones of shape (3, 4) :
 [[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]


...Program finished with exit code 0
Press ENTER to exit console.
```

12.Write a program to create a 2D numpy array initialized with random integers

<u>Source code:</u>

import numpy as np

def random_int_array(rows, cols, low=0, high=10):

return np.random.randint(low, high, size=(rows, cols))

rows = 3

cols = 4

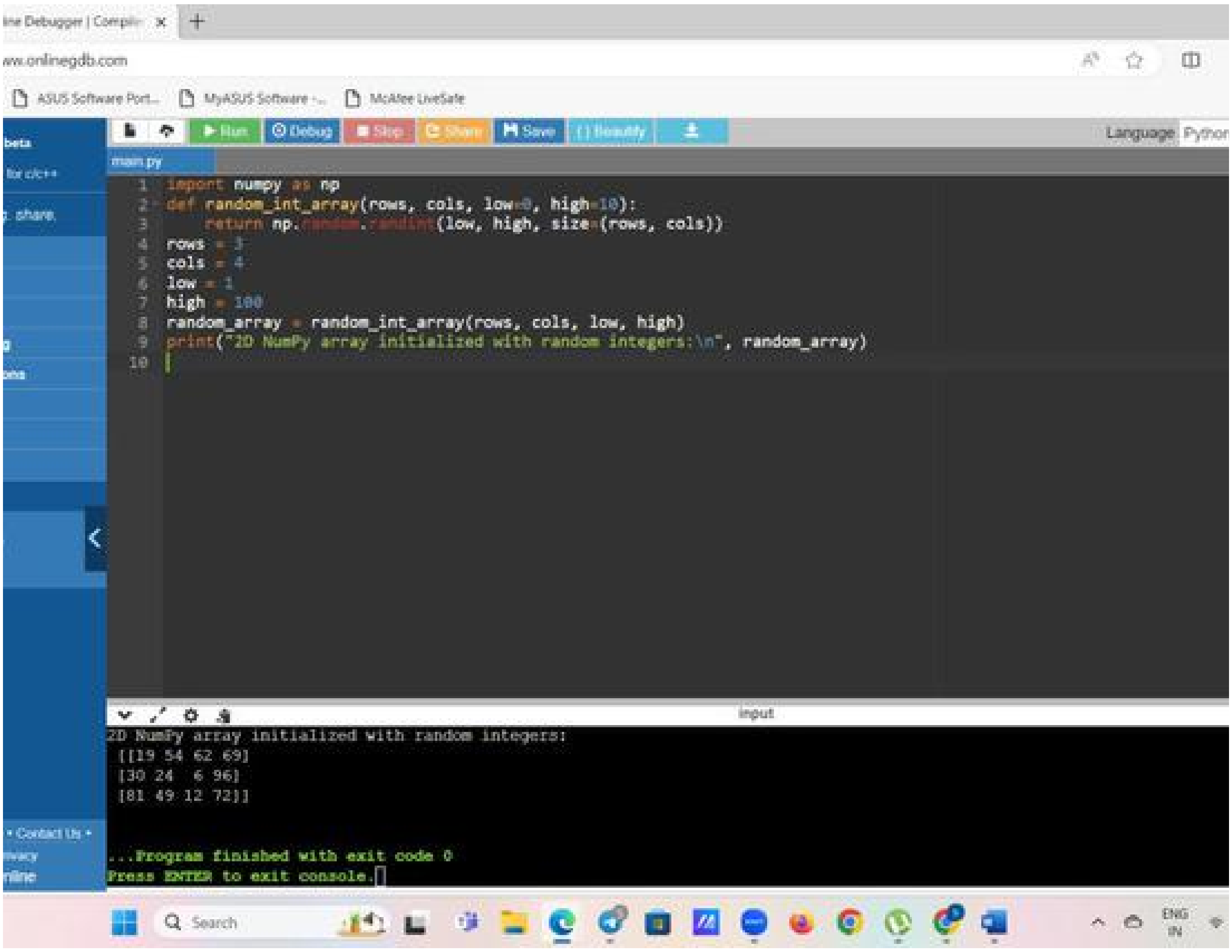
low = 1

high = 100

random_array = random_int_array(rows, cols, low, high)

print("2D NumPy array initialized with random integers:\n", random_array)

Output:



```
import numpy as np
def random_int_array(rows, cols, low=0, high=10):
    return np.random.randint(low, high, size=(rows, cols))
rows = 3
cols = 4
low = 1
high = 100
random_array = random_int_array(rows, cols, low, high)
print("2D NumPy array initialized with random integers:\n", random_array)
```

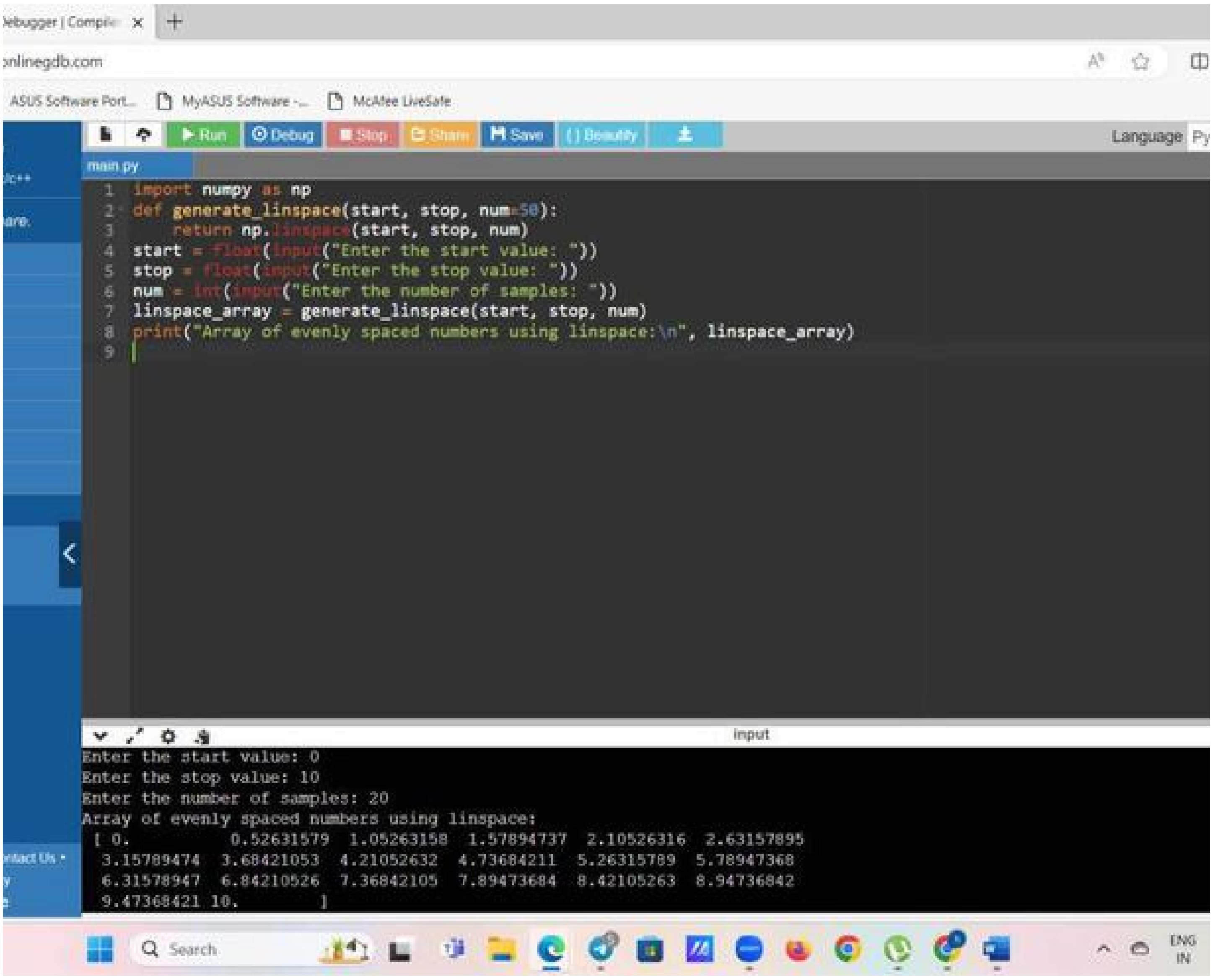
```
2D NumPy array initialized with random integers:
 [[19 54 62 69]
 [30 24  6 96]
 [81 49 12 72]]


...Program finished with exit code 0
Press ENTER to exit console.
```

13.Write a Python program to generate an array of evenly spaced numbers over a specified range using linspace.

Source code:

```
import numpy as np


def generate_linspace(start, stop, num=50):


return np.linspace(start, stop, num)


start = float(input("Enter the start value: "))


stop = float(input("Enter the stop value: "))


num = int(input("Enter the number of samples: "))


linspace_array = generate_linspace(start, stop, num)


print("Array of evenly spaced numbers using linspace:\n", linspace_array)
```

Output:



```python
import numpy as np
def generate_linspace(start, stop, num=50):
    return np.linspace(start, stop, num)
start = float(input("Enter the start value: "))
stop = float(input("Enter the stop value: "))
num = int(input("Enter the number of samples: "))
linspace_array = generate_linspace(start, stop, num)
print("Array of evenly spaced numbers using linspace:\n", linspace_array)
```

```
Enter the start value: 0
Enter the stop value: 10
Enter the number of samples: 20
Array of evenly spaced numbers using linspace:
 [ 0.          0.52631579  1.05263158  1.57894737  2.10526316  2.63157895
  3.15789474  3.68421053  4.21052632  4.73684211  5.26315789  5.78947368
  6.31578947  6.84210526  7.36842105  7.89473684  8.42105263  8.94736842
  9.47368421 10.        ]
```

14.Write a program to generate an array of 10 equally spaced values between 1 and 100 using linspace.

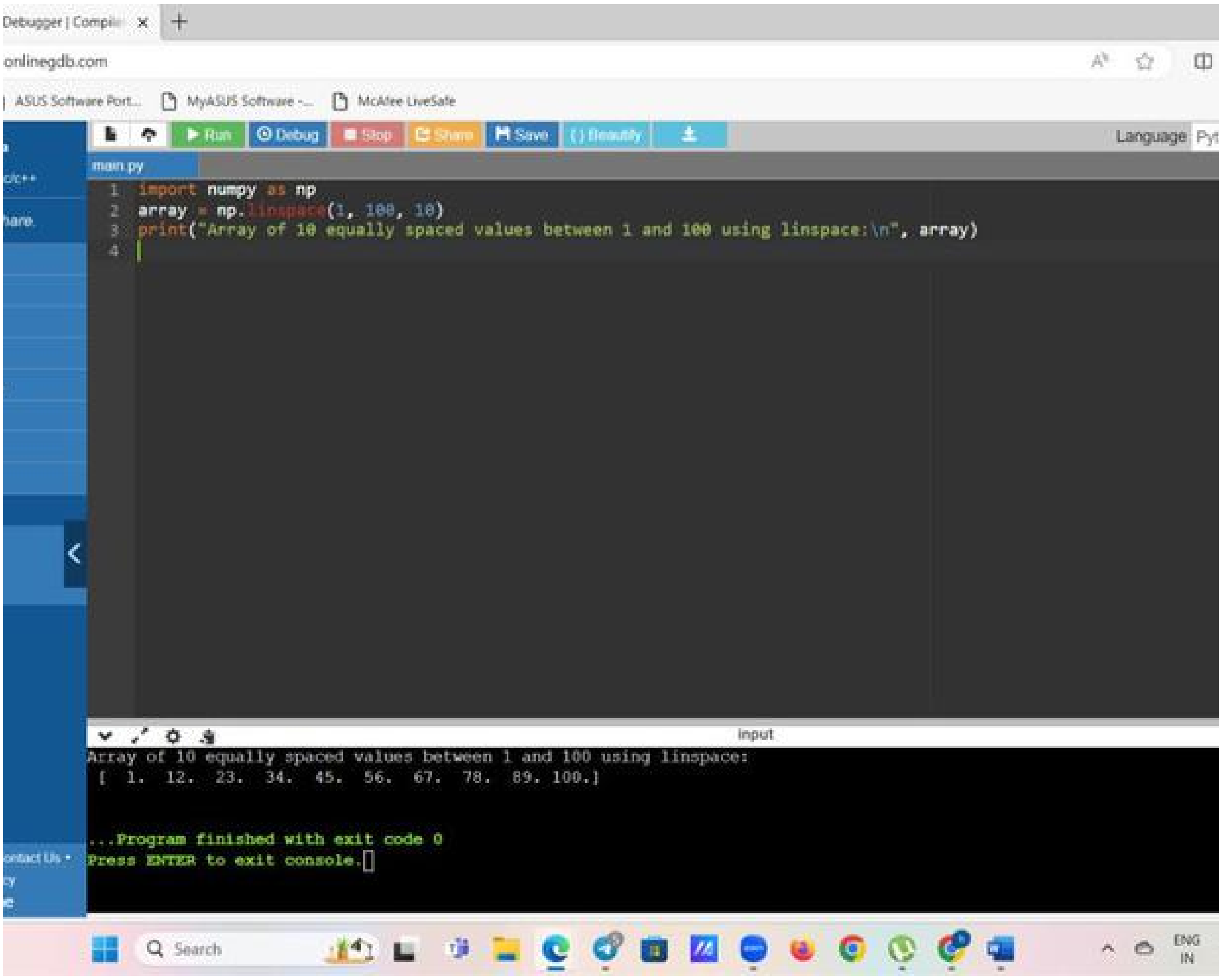
Source code:

import numpy as np

array = np.linspace(1, 100, 10)

print("Array of 10 equally spaced values between 1 and 100 using linspace:\n", array)

Output:



```python
import numpy as np
array = np.linspace(1, 100, 10)
print("Array of 10 equally spaced values between 1 and 100 using linspace:\n", array)
```

```
Array of 10 equally spaced values between 1 and 100 using linspace:
 [  1.  12.  23.  34.  45.  56.  67.  78.  89. 100.]


...Program finished with exit code 0
Press ENTER to exit console.
```

15.Write a Python program to create an array containing even numbers from 2 to 20 using arange.

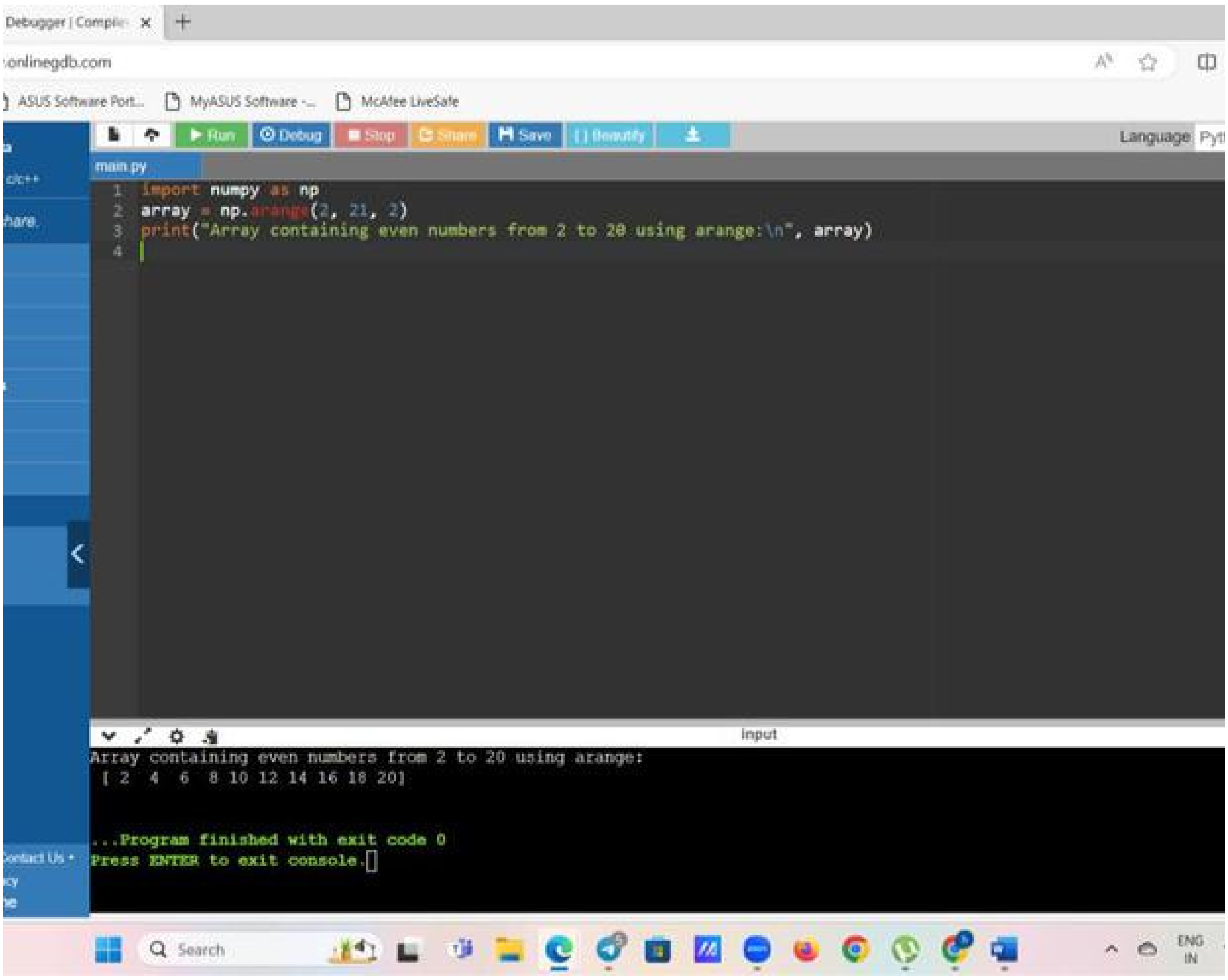
Source code:

import numpy as np

array = np.arange(2, 21, 2)

print("Array containing even numbers from 2 to 20 using arange:\n", array)

Output:



```
import numpy as np
array = np.arange(2, 21, 2)
print("Array containing even numbers from 2 to 20 using arange:\n", array)
```

```
Array containing even numbers from 2 to 20 using arange:
 [ 2  4  6  8 10 12 14 16 18 20]


...Program finished with exit code 0
Press ENTER to exit console.
```

16.Write a program to create an array containing numbers from 1 to 10 with a step size of 0.5 using arrange.

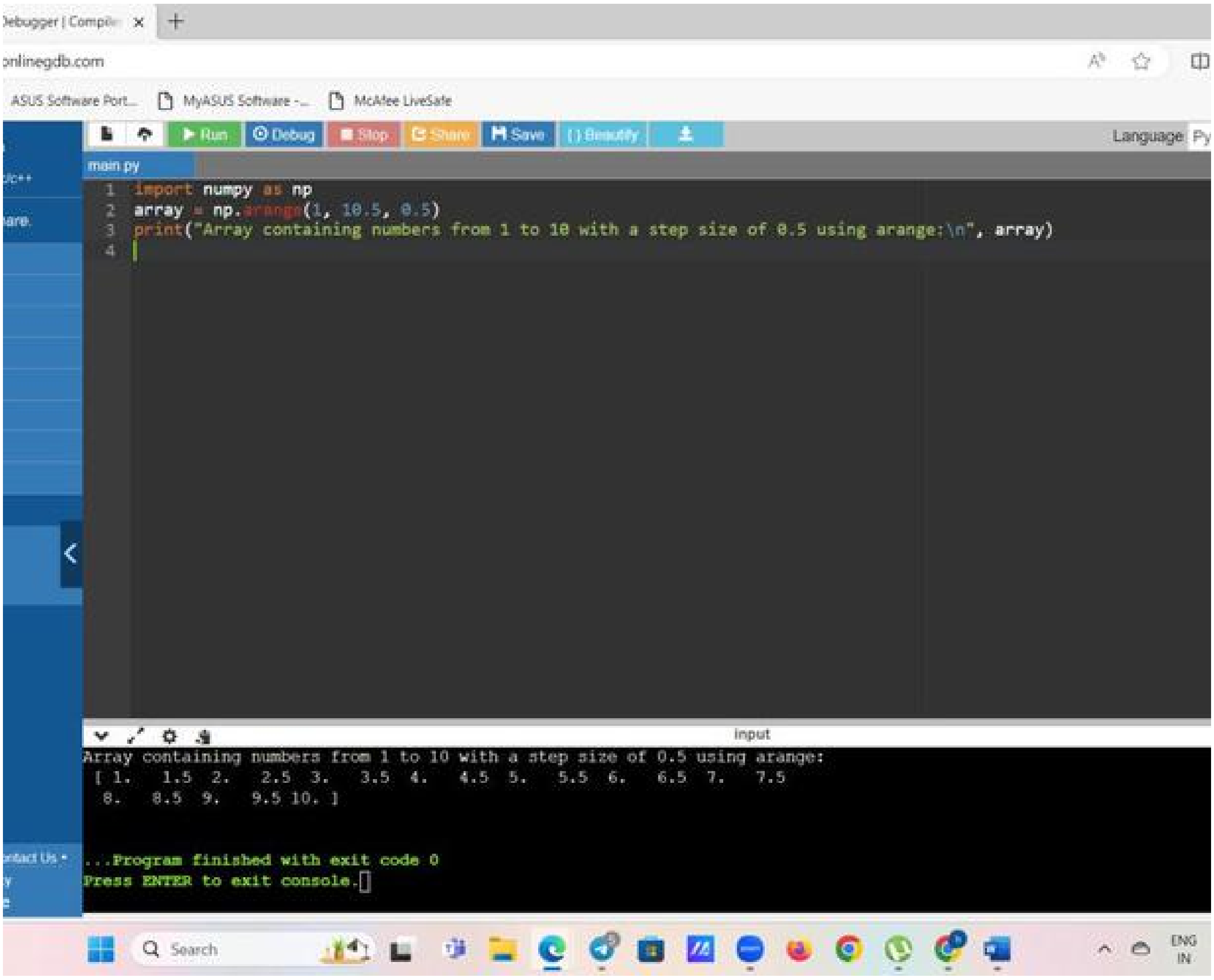
Source code:

import numpy as np

array = np.arange(1, 10.5, 0.5)

print("Array containing numbers from 1 to 10 with a step size of 0.5 using arange:\n", array)

Output:



```python
import numpy as np
array = np.arange(1, 10.5, 0.5)
print("Array containing numbers from 1 to 10 with a step size of 0.5 using arange:\n", array)
```

```
Array containing numbers from 1 to 10 with a step size of 0.5 using arange:
 [ 1.   1.5  2.   2.5  3.   3.5  4.   4.5  5.   5.5  6.   6.5  7.   7.5
  8.   8.5  9.   9.5 10. ]


...Program finished with exit code 0
Press ENTER to exit console.
```