

1.



```
main.py  [Icons] Save Run Shell Clear
1 def calculate_rectangle_area(length, width):
2     area = length * width
3     return area
4
5 # Taking input from the user
6 length = float(input("Enter the length of the rectangle: "))
7 width = float(input("Enter the width of the rectangle: "))
8
9 # Calculating the area
10 area = calculate_rectangle_area(length, width)
11
12 # Displaying the result
13 print(f"The area of the rectangle with length {length} and width {width} is: {area}")
14
```

Enter the length of the rectangle: 5  
Enter the width of the rectangle: 4  
The area of the rectangle with length 5.0 and width 4.0 is: 20.0  
> |

2.



```
main.py  [Icons] Save Run Shell Clear
1 def miles_to_kilometers(miles):
2     kilometers = miles * 1.60934
3     return kilometers
4
5 # Taking input from the user
6 miles = float(input("Enter the distance in miles: "))
7
8 # Converting miles to kilometers
9 kilometers = miles_to_kilometers(miles)
10
11 # Displaying the result
12 print(f"{miles} miles is equal to {kilometers:.2f} kilometers")
13
```

Enter the distance in miles: 8  
8.0 miles is equal to 12.87 kilometers  
> |

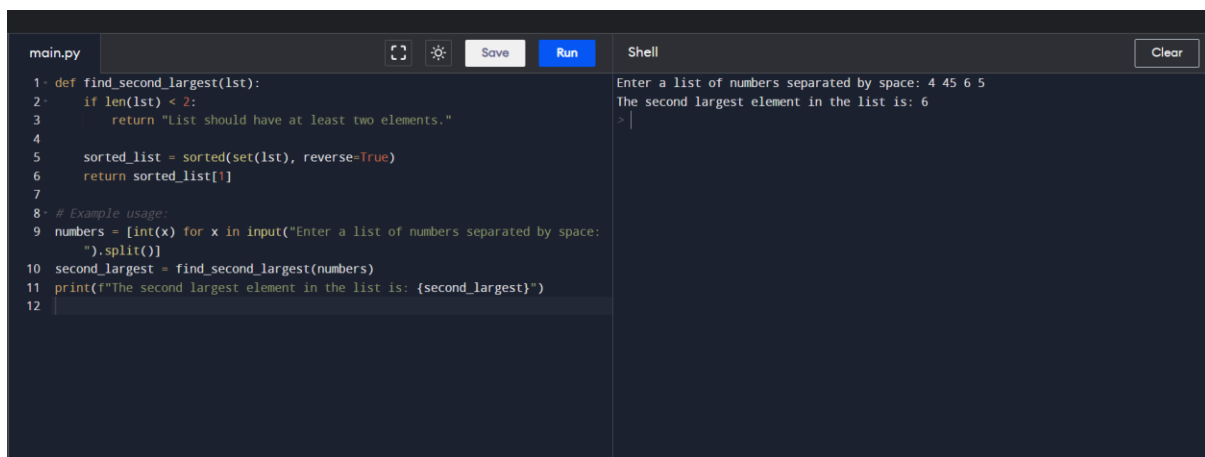
3.



```
main.py  [Icons] Save Run Shell Clear
1 def is_palindrome(s):
2     s = s.lower() # Convert the string to lowercase for case-insensitive comparison
3     return s == s[::-1]
4
5 # Example usage:
6 input_string = input("Enter a string to check if it's a palindrome: ")
7 if is_palindrome(input_string):
8     print(f"{input_string} is a palindrome.")
9 else:
10    print(f"{input_string} is not a palindrome.")
11
```

Enter a string to check if it's a palindrome: 454  
454 is a palindrome.  
> |

4.



```
main.py  [Icons] Save Run Shell Clear
1 def find_second_largest(lst):
2     if len(lst) < 2:
3         return "List should have at least two elements."
4
5     sorted_list = sorted(set(lst), reverse=True)
6     return sorted_list[1]
7
8 # Example usage:
9 numbers = [int(x) for x in input("Enter a list of numbers separated by space: ").split()]
10 second_largest = find_second_largest(numbers)
11 print(f"The second largest element in the list is: {second_largest}")
12
```

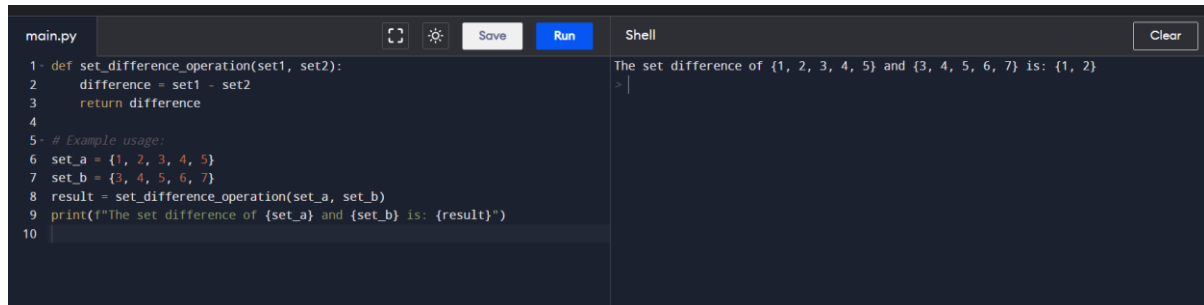
Enter a list of numbers separated by space: 4 45 6 5  
The second largest element in the list is: 6  
> |

5.

In Python, indentation is used to define the block of code. Unlike many other programming languages that use braces {}, Python uses indentation to indicate the scope of control structures,

functions, classes, and other code blocks. The standard convention is to use four spaces for each level of indentation. Incorrect indentation can lead to syntax errors or unexpected behavior.

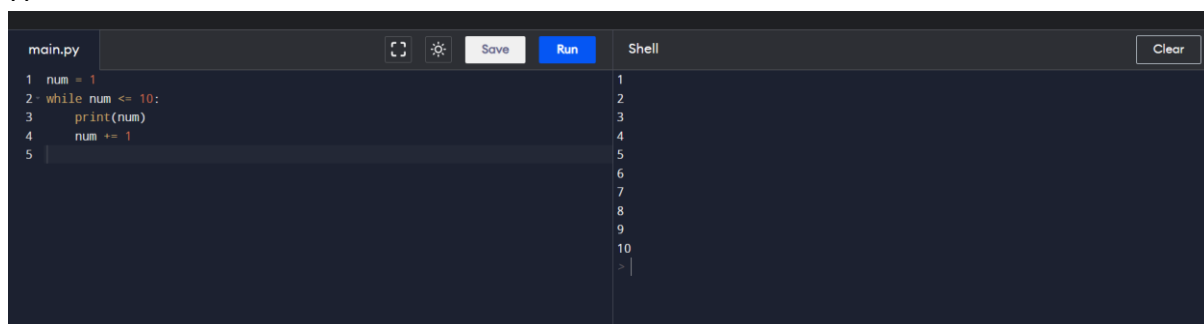
6.



```
main.py  [Icons] Save Run Shell Clear
1 def set_difference_operation(set1, set2):
2     difference = set1 - set2
3     return difference
4
5 # Example usage:
6 set_a = {1, 2, 3, 4, 5}
7 set_b = {3, 4, 5, 6, 7}
8 result = set_difference_operation(set_a, set_b)
9 print(f"The set difference of {set_a} and {set_b} is: {result}")
10
```

The set difference of {1, 2, 3, 4, 5} and {3, 4, 5, 6, 7} is: {1, 2}

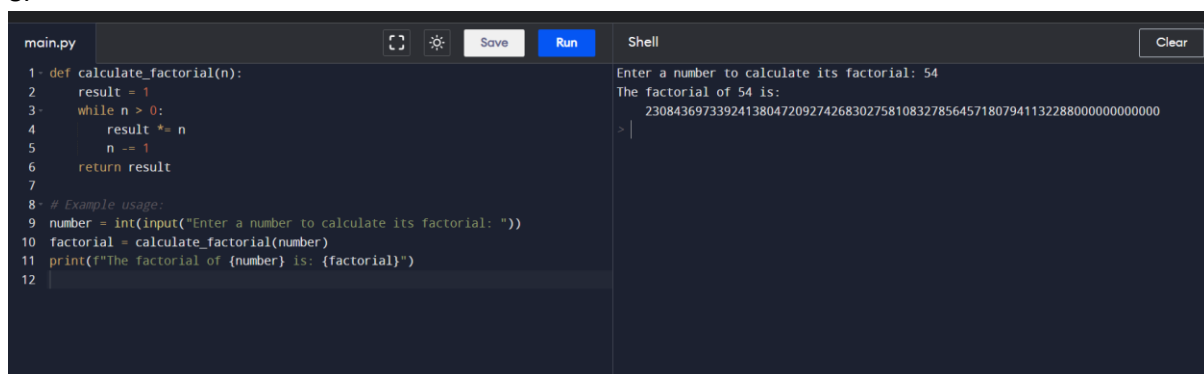
7.



```
main.py  [Icons] Save Run Shell Clear
1 num = 1
2 while num <= 10:
3     print(num)
4     num += 1
5
```

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
>

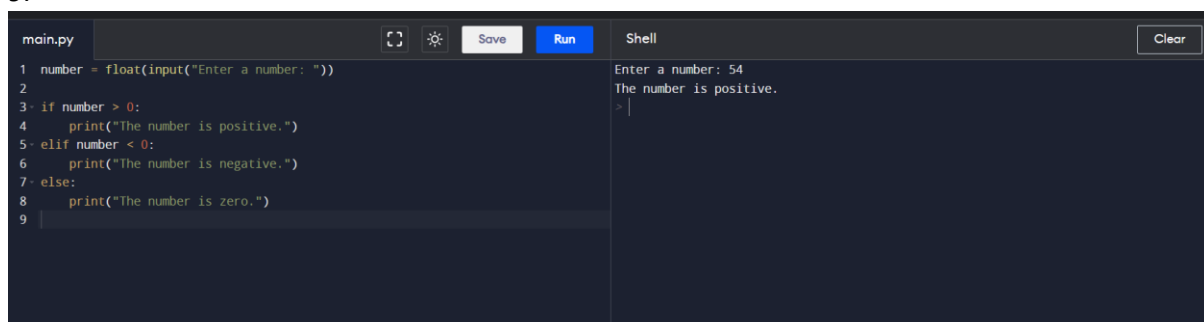
8.



```
main.py  [Icons] Save Run Shell Clear
1 def calculate_factorial(n):
2     result = 1
3     while n > 0:
4         result *= n
5         n -= 1
6     return result
7
8 # Example usage:
9 number = int(input("Enter a number to calculate its factorial: "))
10 factorial = calculate_factorial(number)
11 print(f"The factorial of {number} is: {factorial}")
12
```

Enter a number to calculate its factorial: 54  
The factorial of 54 is:  
230843697339241380472092742683027581083278564571807941132288000000000000  
>

9.



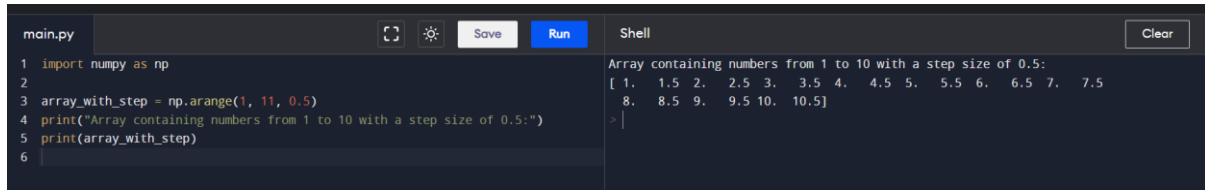
```
main.py  [Icons] Save Run Shell Clear
1 number = float(input("Enter a number: "))
2
3 if number > 0:
4     print("The number is positive.")
5 elif number < 0:
6     print("The number is negative.")
7 else:
8     print("The number is zero.")
9
```

Enter a number: 54  
The number is positive.  
>

10.



16.



The screenshot shows a Jupyter Notebook interface with a dark theme. The left pane displays a Python script in a file named `main.py`. The script imports `numpy` as `np`, creates an array `array_with_step` using `np.arange(1, 11, 0.5)`, and prints a message and the array. The right pane, labeled 'Shell', shows the output of the script, which is a formatted array of numbers from 1 to 10 with a step size of 0.5. The interface includes icons for file operations and buttons for 'Save' and 'Run'.

```
main.py
1 import numpy as np
2
3 array_with_step = np.arange(1, 11, 0.5)
4 print("Array containing numbers from 1 to 10 with a step size of 0.5:")
5 print(array_with_step)
6
```

Shell

Array containing numbers from 1 to 10 with a step size of 0.5:

```
[ 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. 10.5]
```