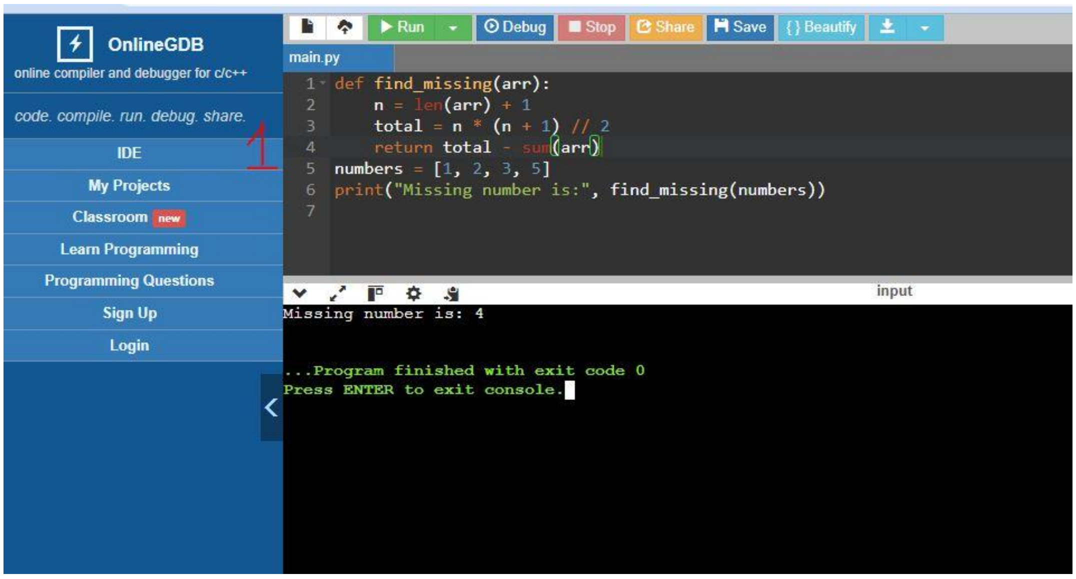
Python Developer

Task:4

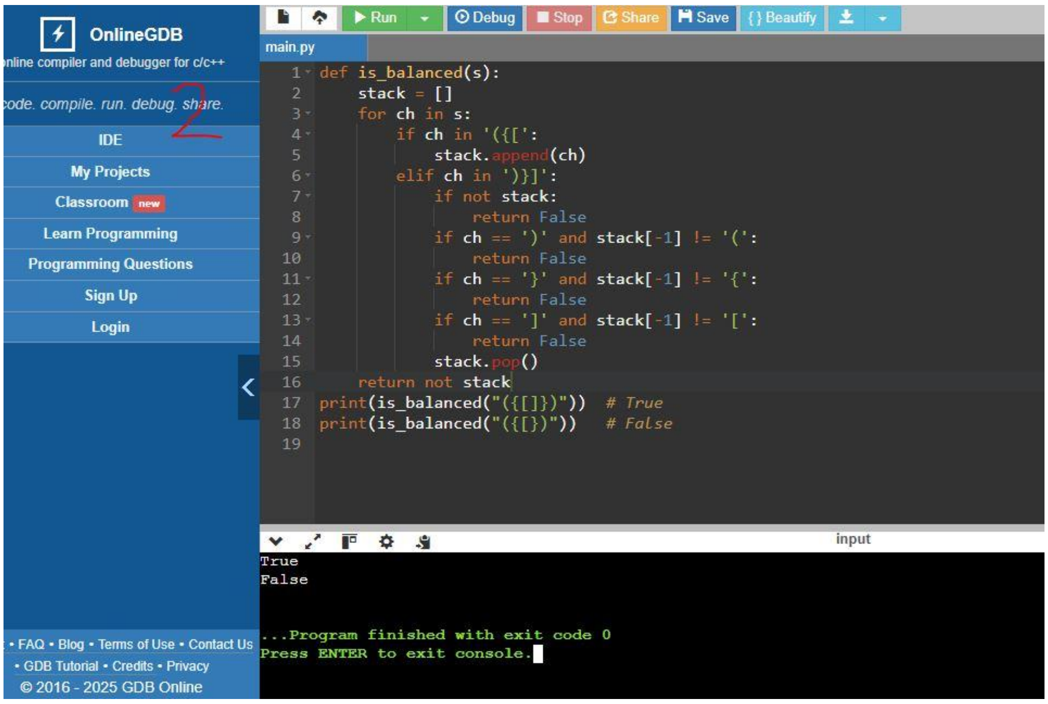
25. Find Missing Number



```python
def find_missing(arr):
    n = len(arr) + 1
    total = n * (n + 1) // 2
    return total - sum(arr)
numbers = [1, 2, 3, 5]
print("Missing number is:", find_missing(numbers))
```

```
Missing number is: 4


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

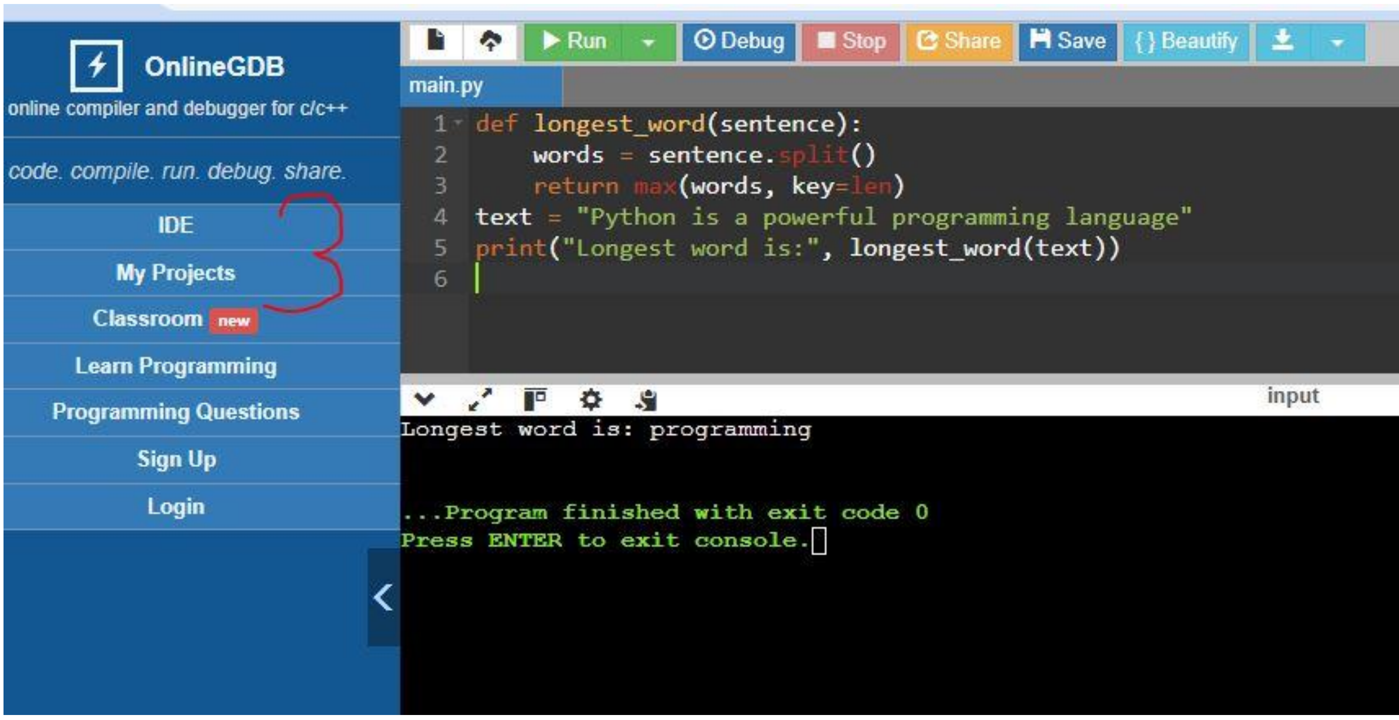
26. Check Balanced Parentheses



```python
def is_balanced(s):
    stack = []
    for ch in s:
        if ch in '({[':
            stack.append(ch)
        elif ch in ')}]':
            if not stack:
                return False
            if ch == ')' and stack[-1] != '(':
                return False
            if ch == '}' and stack[-1] != '{':
                return False
            if ch == ']' and stack[-1] != '[':
                return False
            stack.pop()
    return not stack
print(is_balanced("({[]})"))  # True
print(is_balanced("({[})"))   # False
```

```
True
False


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

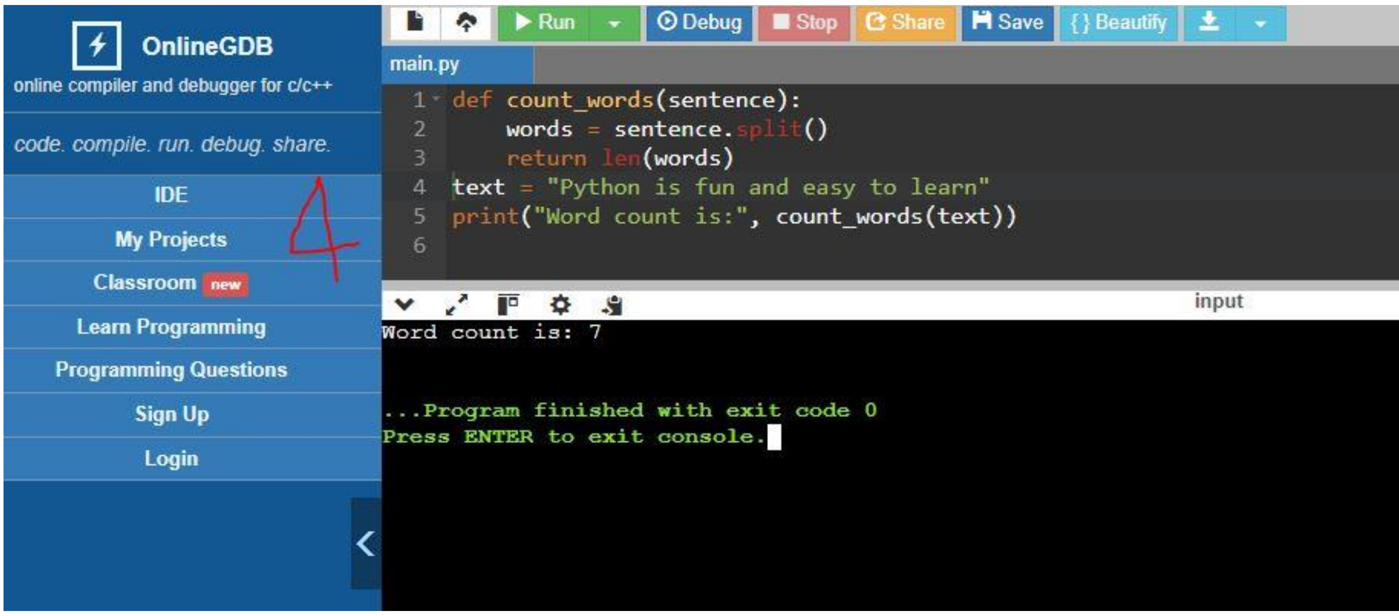
27. Longest Word in a Sentence



```python
def longest_word(sentence):
    words = sentence.split()
    return max(words, key=len)
text = "Python is a powerful programming language"
print("Longest word is:", longest_word(text))
```

```
Longest word is: programming


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

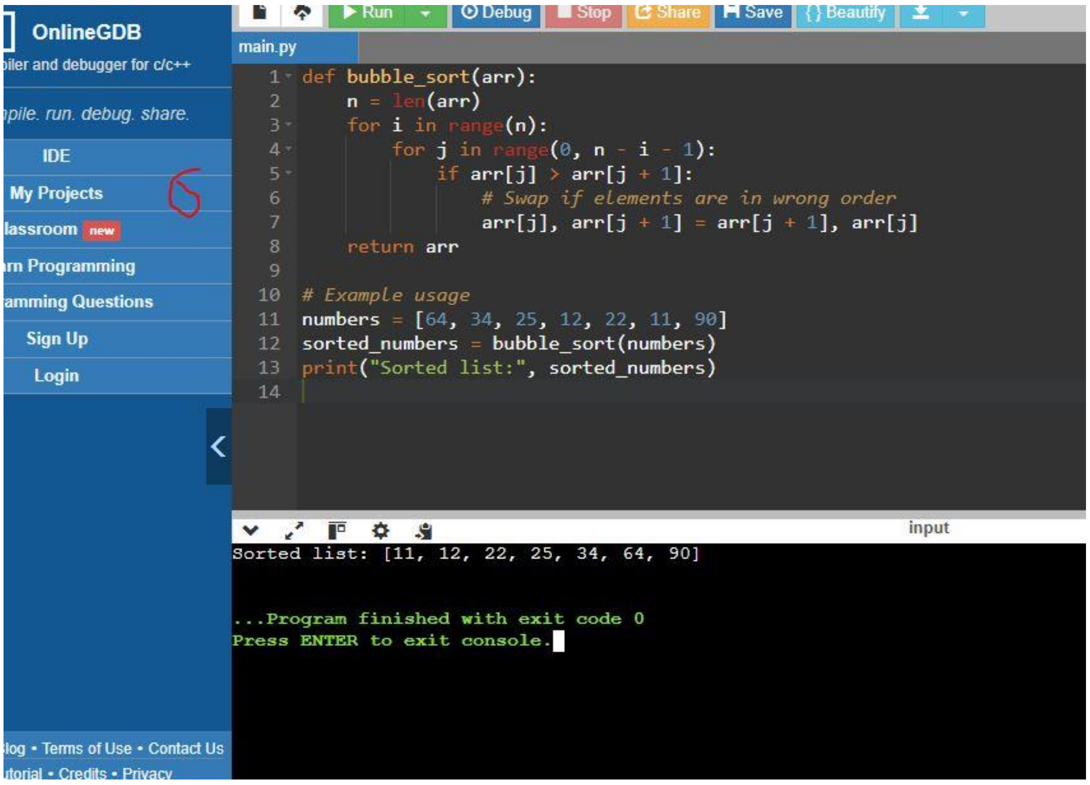
28. Count Words in a Sentence



```python
def count_words(sentence):
    words = sentence.split()
    return len(words)
text = "Python is fun and easy to learn"
print("Word count is:", count_words(text))
```

```
Word count is: 7


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

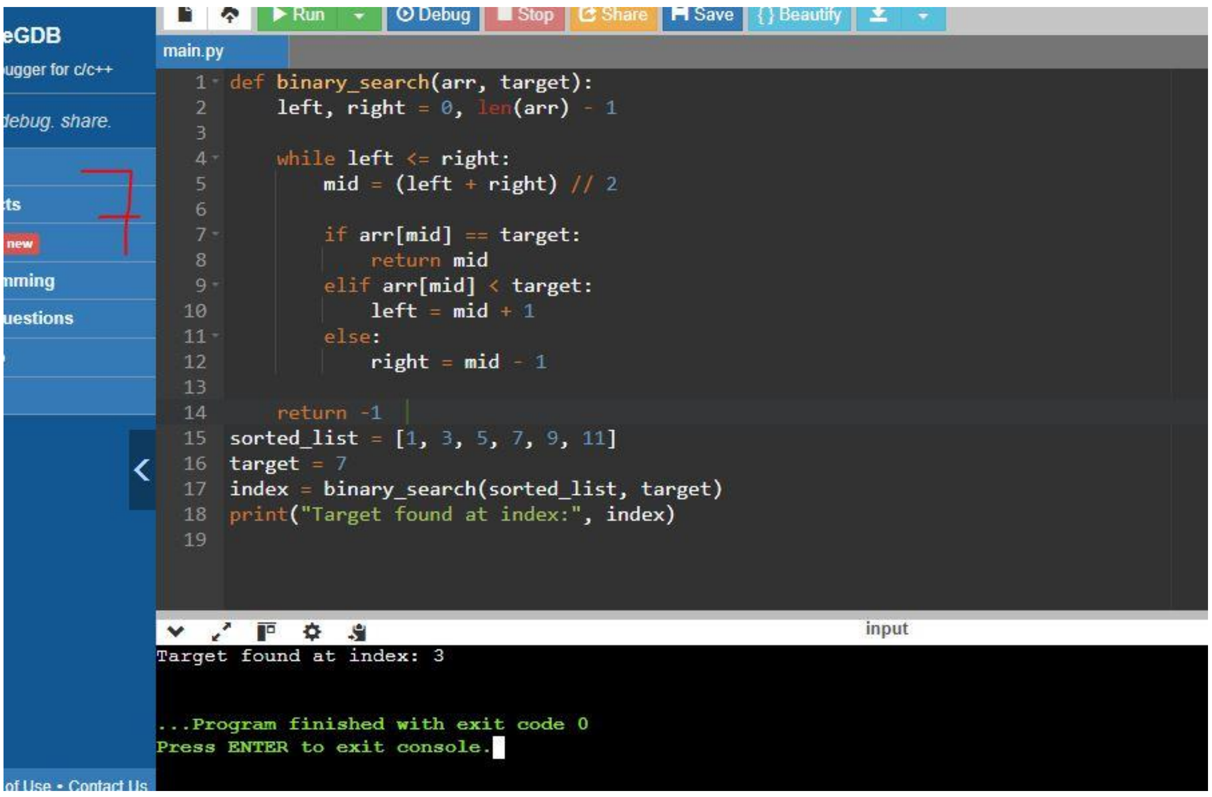
29. Check Pythagorean Triplet



```python
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                # Swap if elements are in wrong order
                arr[j], arr[j + 1] = arr[j + 1], arr[j]
    return arr

# Example usage
numbers = [64, 34, 25, 12, 22, 11, 90]
sorted_numbers = bubble_sort(numbers)
print("Sorted list:", sorted_numbers)
```

```
Sorted list: [11, 12, 22, 25, 34, 64, 90]


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

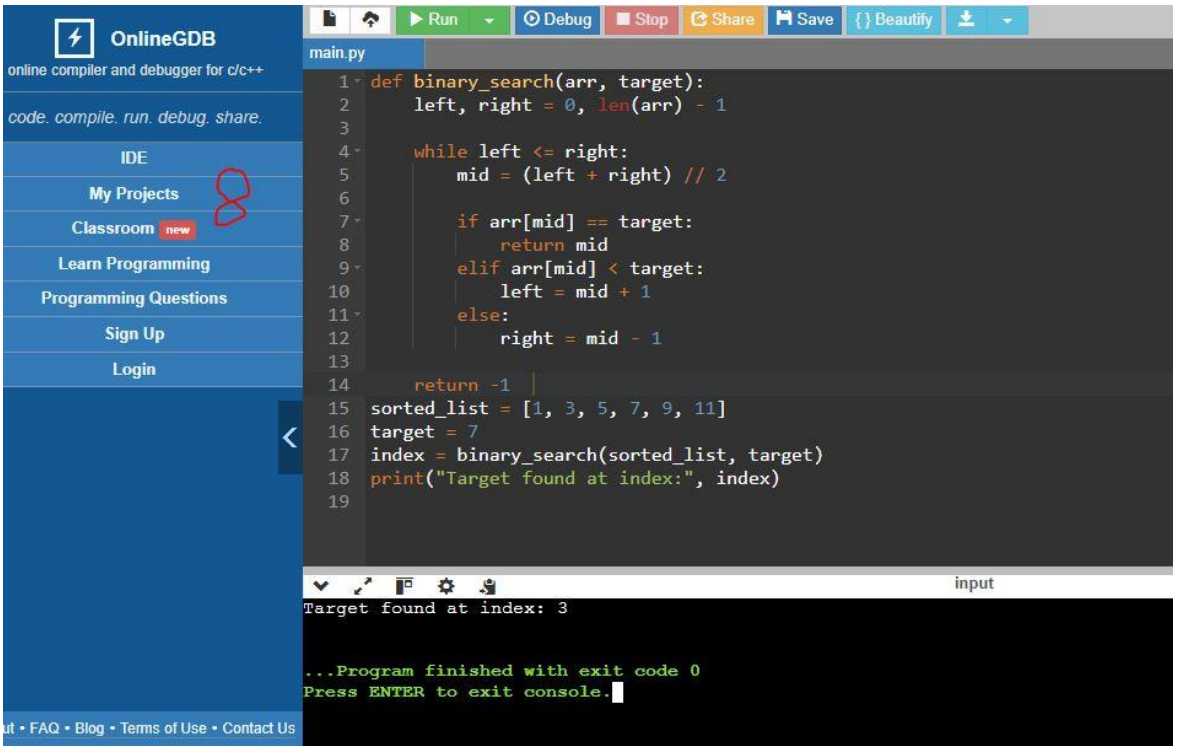
30. Bubble Sort



```python
def binary_search(arr, target):
    left, right = 0, len(arr) - 1

    while left <= right:
        mid = (left + right) // 2

        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1
sorted_list = [1, 3, 5, 7, 9, 11]
target = 7
index = binary_search(sorted_list, target)
print("Target found at index:", index)
```

```
Target found at index: 3


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

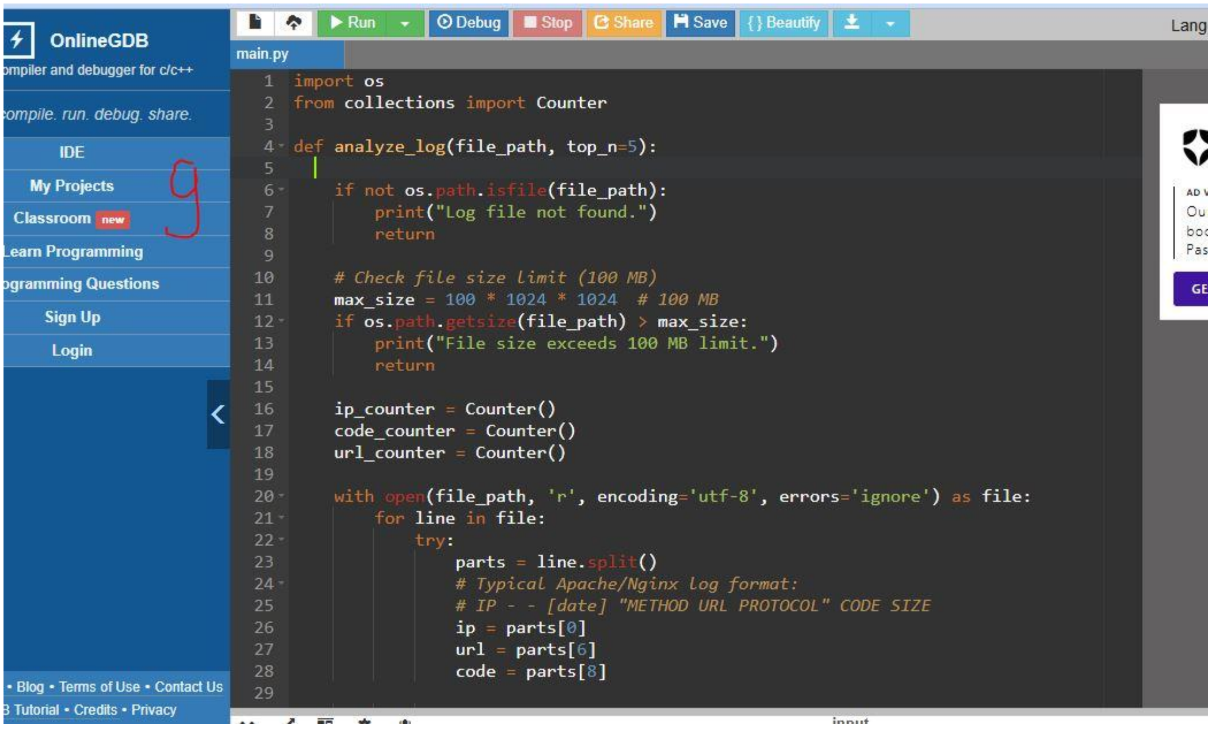
31. Binary Search



```python
def binary_search(arr, target):
    left, right = 0, len(arr) - 1

    while left <= right:
        mid = (left + right) // 2

        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1
sorted_list = [1, 3, 5, 7, 9, 11]
target = 7
index = binary_search(sorted_list, target)
print("Target found at index:", index)
```

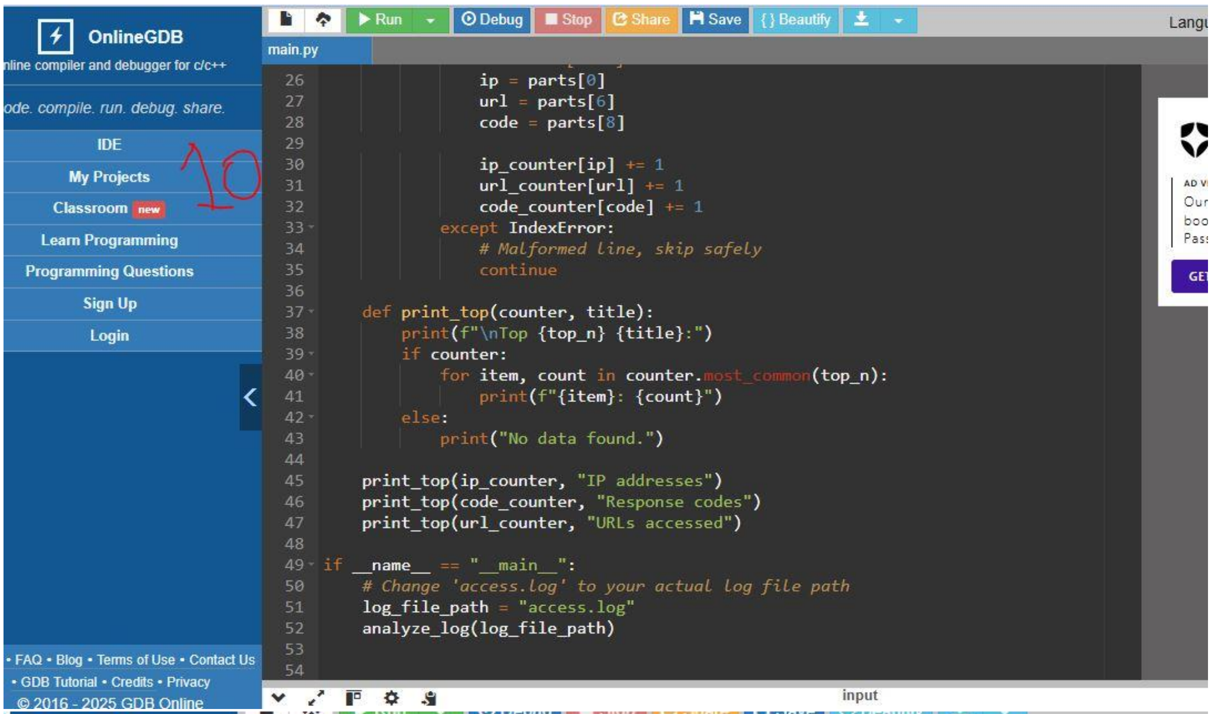
```
Target found at index: 3


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

32. Find Subarray with Given Sum

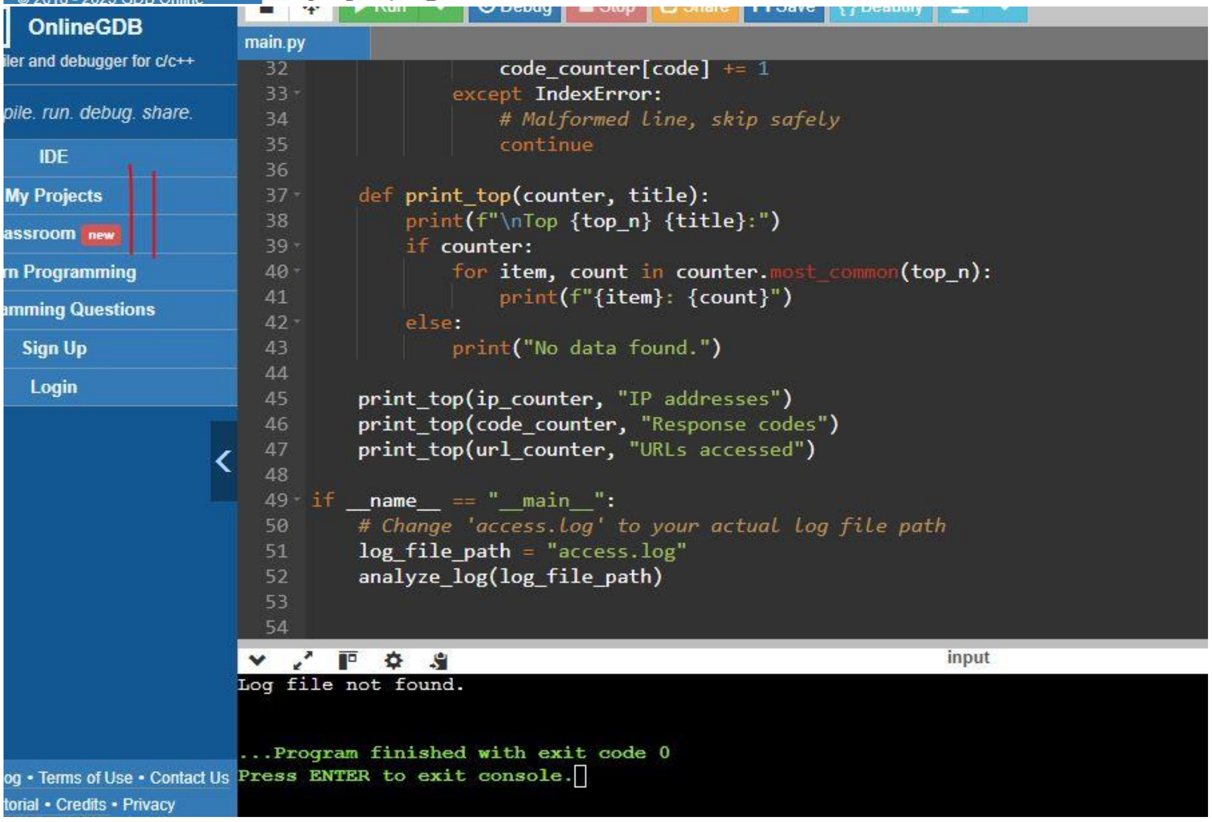


```python
import os
from collections import Counter

def analyze_log(file_path, top_n=5):

    if not os.path.isfile(file_path):
        print("Log file not found.")
        return

    # Check file size limit (100 MB)
    max_size = 100 * 1024 * 1024  # 100 MB
    if os.path.getsize(file_path) > max_size:
        print("File size exceeds 100 MB limit.")
        return

    ip_counter = Counter()
    code_counter = Counter()
    url_counter = Counter()

    with open(file_path, 'r', encoding='utf-8', errors='ignore') as file:
        for line in file:
            try:
                parts = line.split()
                # Typical Apache/Nginx log format:
                # IP - - [date] "METHOD URL PROTOCOL" CODE SIZE
                ip = parts[0]
                url = parts[6]
                code = parts[8]
```

4. Log Analysis System:



```
                ip = parts[0]
                url = parts[6]
                code = parts[8]

                ip_counter[ip] += 1
                url_counter[url] += 1
                code_counter[code] += 1
            except IndexError:
                # Malformed line, skip safely
                continue

    def print_top(counter, title):
        print(f"\nTop {top_n} {title}:")
        if counter:
            for item, count in counter.most_common(top_n):
                print(f"{item}: {count}")
        else:
            print("No data found.")

    print_top(ip_counter, "IP addresses")
    print_top(code_counter, "Response codes")
    print_top(url_counter, "URLs accessed")

if __name__ == "__main__":
    # Change 'access.log' to your actual log file path
    log_file_path = "access.log"
    analyze_log(log_file_path)
```



```
                code_counter[code] += 1
            except IndexError:
                # Malformed line, skip safely
                continue

    def print_top(counter, title):
        print(f"\nTop {top_n} {title}:")
        if counter:
            for item, count in counter.most_common(top_n):
                print(f"{item}: {count}")
        else:
            print("No data found.")

    print_top(ip_counter, "IP addresses")
    print_top(code_counter, "Response codes")
    print_top(url_counter, "URLs accessed")

if __name__ == "__main__":
    # Change 'access.log' to your actual log file path
    log_file_path = "access.log"
    analyze_log(log_file_path)
```

```
Log file not found.


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