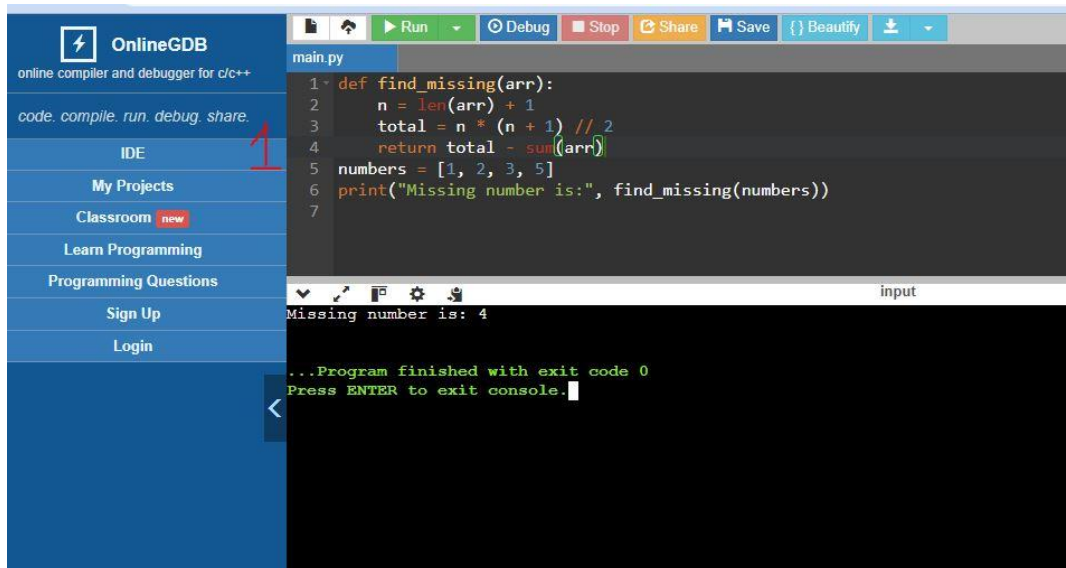


# Python Developer

## Task:4

### 25. Find Missing Number



OnlineGDB  
online compiler and debugger for c/c++  
code. compile. run. debug. share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

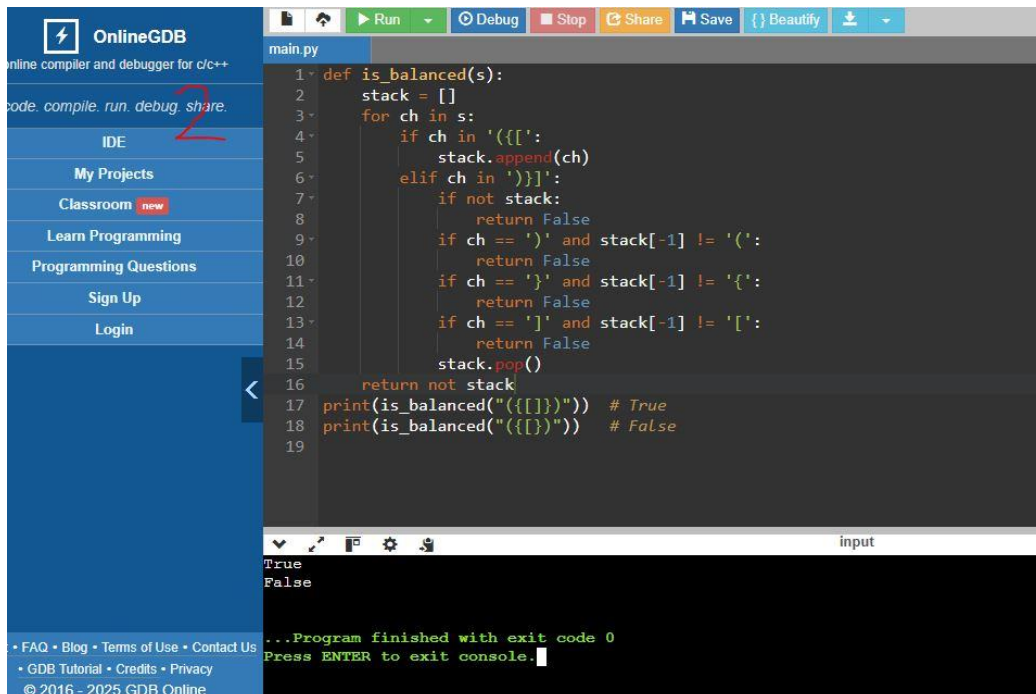
```
main.py
1 def find_missing(arr):
2     n = len(arr) + 1
3     total = n * (n + 1) // 2
4     return total - sum(arr)
5 numbers = [1, 2, 3, 5]
6 print("Missing number is:", find_missing(numbers))
7
```

input

Missing number is: 4

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

### 26. Check Balanced Parentheses



OnlineGDB  
online compiler and debugger for c/c++  
code. compile. run. debug. share.

IDE

My Projects

Classroom new

Learn Programming

Programming Questions

Sign Up

Login

```
main.py
1 def is_balanced(s):
2     stack = []
3     for ch in s:
4         if ch in '({[':
5             stack.append(ch)
6         elif ch in ')}]':
7             if not stack:
8                 return False
9             if ch == ')' and stack[-1] != '(':
10                return False
11             if ch == '}' and stack[-1] != '{':
12                return False
13             if ch == ']' and stack[-1] != '[':
14                return False
15             stack.pop()
16     return not stack
17 print(is_balanced("({[]})")) # True
18 print(is_balanced("({[]})")) # False
19
```

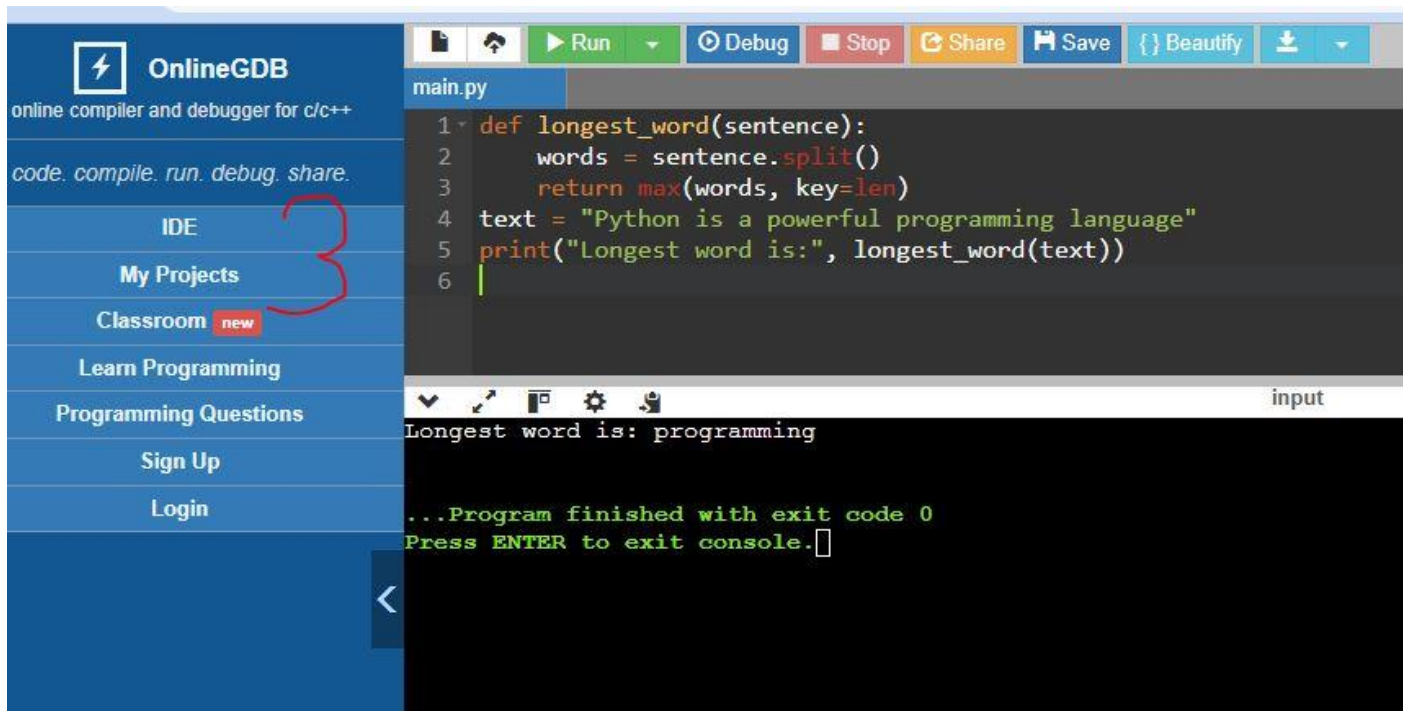
input

True  
False

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

• FAQ • Blog • Terms of Use • Contact Us  
• GDB Tutorial • Credits • Privacy  
© 2016 - 2025 GDB Online

## 27. Longest Word in a Sentence



The screenshot shows the OnlineGDB IDE interface. On the left is a sidebar with navigation links: IDE, My Projects, Classroom (marked with a red '3'), Learn Programming, Programming Questions, Sign Up, and Login. The main editor area displays a Python file named `main.py` with the following code:

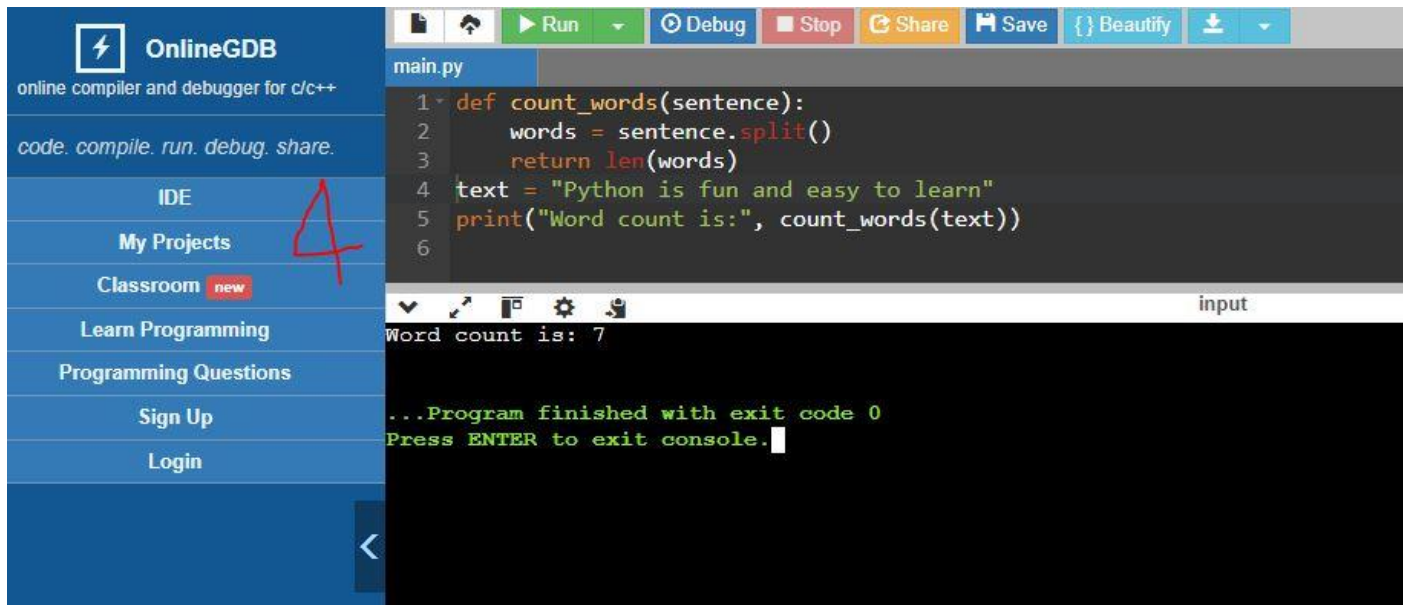
```
1 def longest_word(sentence):
2     words = sentence.split()
3     return max(words, key=len)
4 text = "Python is a powerful programming language"
5 print("Longest word is:", longest_word(text))
6
```

Below the code editor, the console output shows the result of running the program:

```
input
Longest word is: programming

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

## 28. Count Words in a Sentence



The screenshot shows the OnlineGDB IDE interface. On the left is a sidebar with navigation links: IDE, My Projects, Classroom (marked with a red '4'), Learn Programming, Programming Questions, Sign Up, and Login. The main editor area displays a Python file named `main.py` with the following code:

```
1 def count_words(sentence):
2     words = sentence.split()
3     return len(words)
4 text = "Python is fun and easy to learn"
5 print("Word count is:", count_words(text))
6
```

Below the code editor, the console output shows the result of running the program:

```
input
Word count is: 7

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

## 29. Check Pythagorean Triplet

```
OnlineGDB
Compiler and debugger for c/c++
Compile. run. debug. share.
IDE
My Projects 5
Classroom new
Learn Programming
Ask Programming Questions
Sign Up
Login

main.py
1 def bubble_sort(arr):
2     n = len(arr)
3     for i in range(n):
4         for j in range(0, n - i - 1):
5             if arr[j] > arr[j + 1]:
6                 # Swap if elements are in wrong order
7                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
8     return arr
9
10 # Example usage
11 numbers = [64, 34, 25, 12, 22, 11, 90]
12 sorted_numbers = bubble_sort(numbers)
13 print("Sorted list:", sorted_numbers)
14

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

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

## 30. Bubble Sort

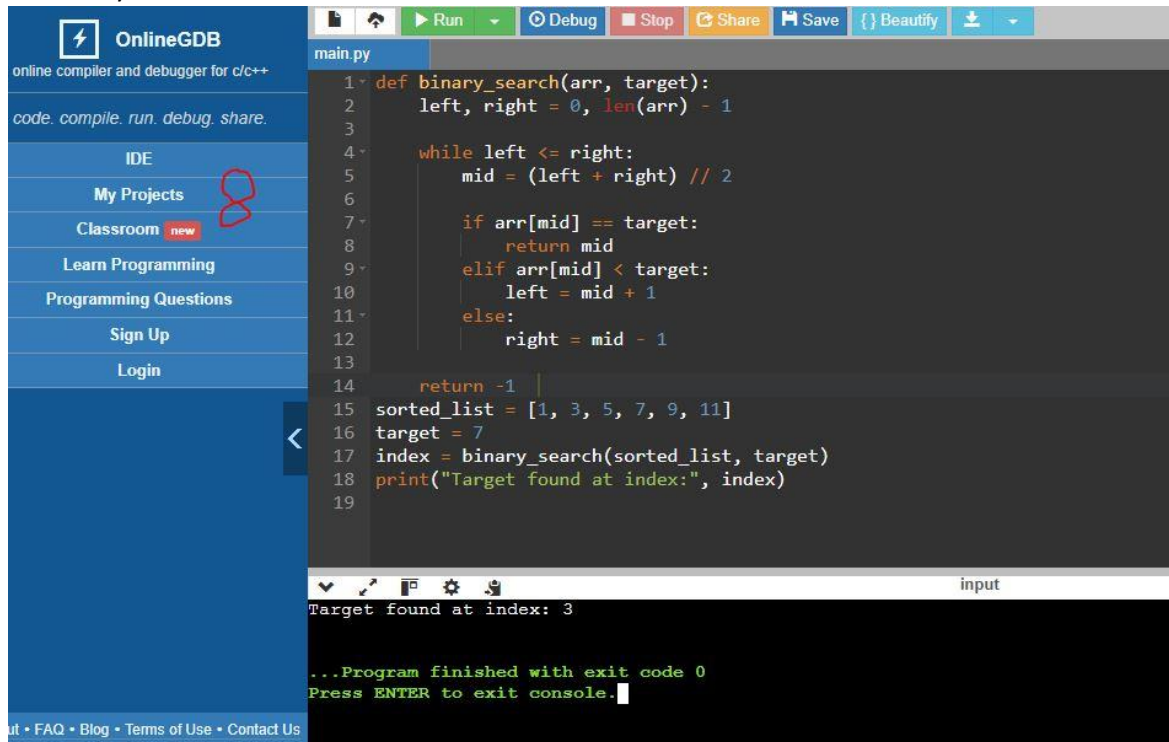
```
OnlineGDB
Compiler and debugger for c/c++
Compile. run. debug. share.
IDE
My Projects 7
Classroom new
Learn Programming
Ask Programming Questions
Sign Up
Login

main.py
1 def binary_search(arr, target):
2     left, right = 0, len(arr) - 1
3
4     while left <= right:
5         mid = (left + right) // 2
6
7         if arr[mid] == target:
8             return mid
9         elif arr[mid] < target:
10            left = mid + 1
11        else:
12            right = mid - 1
13
14    return -1
15
16 sorted_list = [1, 3, 5, 7, 9, 11]
17 target = 7
18 index = binary_search(sorted_list, target)
19 print("Target found at index:", index)

input
Target found at index: 3

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

### 31. Binary Search

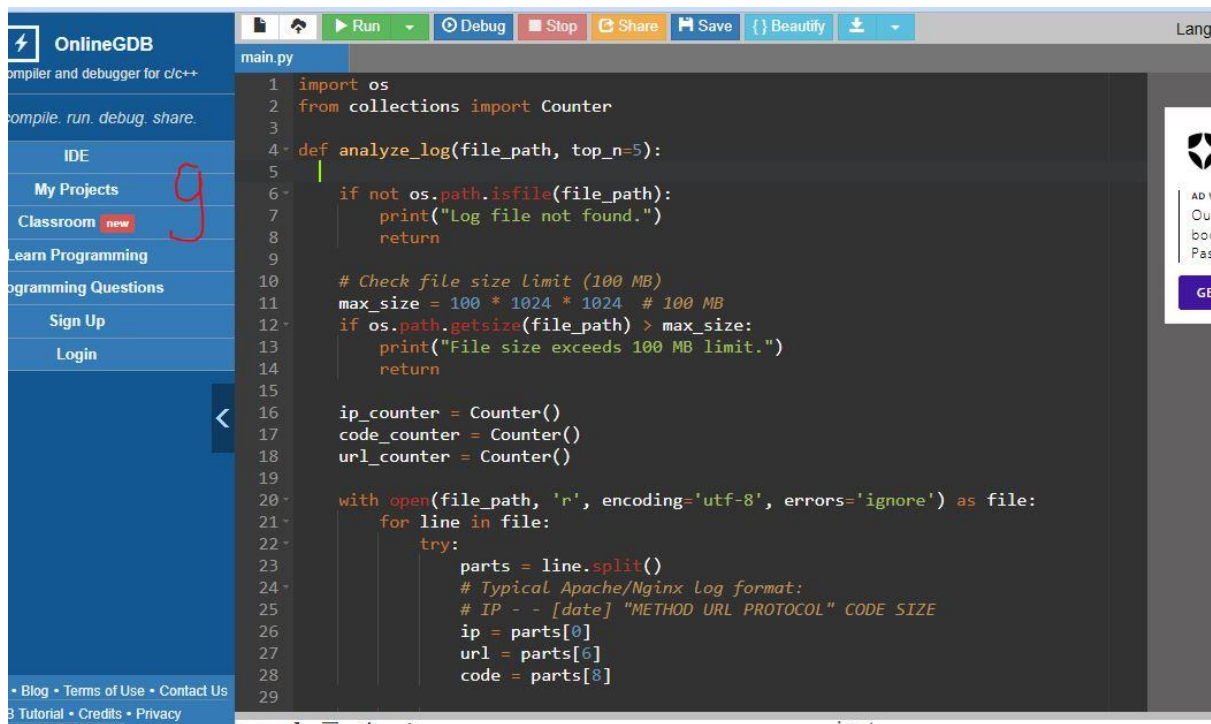


The screenshot shows the OnlineGDB IDE interface. On the left is a sidebar with navigation links: IDE, My Projects (with a red circle around it), Classroom (with a 'new' badge), Learn Programming, Programming Questions, Sign Up, and Login. The main editor area displays a Python file named 'main.py' containing a binary search function. The code is as follows:

```
1 def binary_search(arr, target):
2     left, right = 0, len(arr) - 1
3
4     while left <= right:
5         mid = (left + right) // 2
6
7         if arr[mid] == target:
8             return mid
9         elif arr[mid] < target:
10            left = mid + 1
11        else:
12            right = mid - 1
13
14    return -1
15
16 sorted_list = [1, 3, 5, 7, 9, 11]
17 target = 7
18 index = binary_search(sorted_list, target)
19 print("Target found at index:", index)
```

Below the code editor is a console window showing the output: "Target found at index: 3". At the bottom, a message states "...Program finished with exit code 0" and "Press ENTER to exit console." The top of the IDE features a toolbar with buttons for Run, Debug, Stop, Share, Save, Beautify, and a download icon.

### 32. Find Subarray with Given Sum



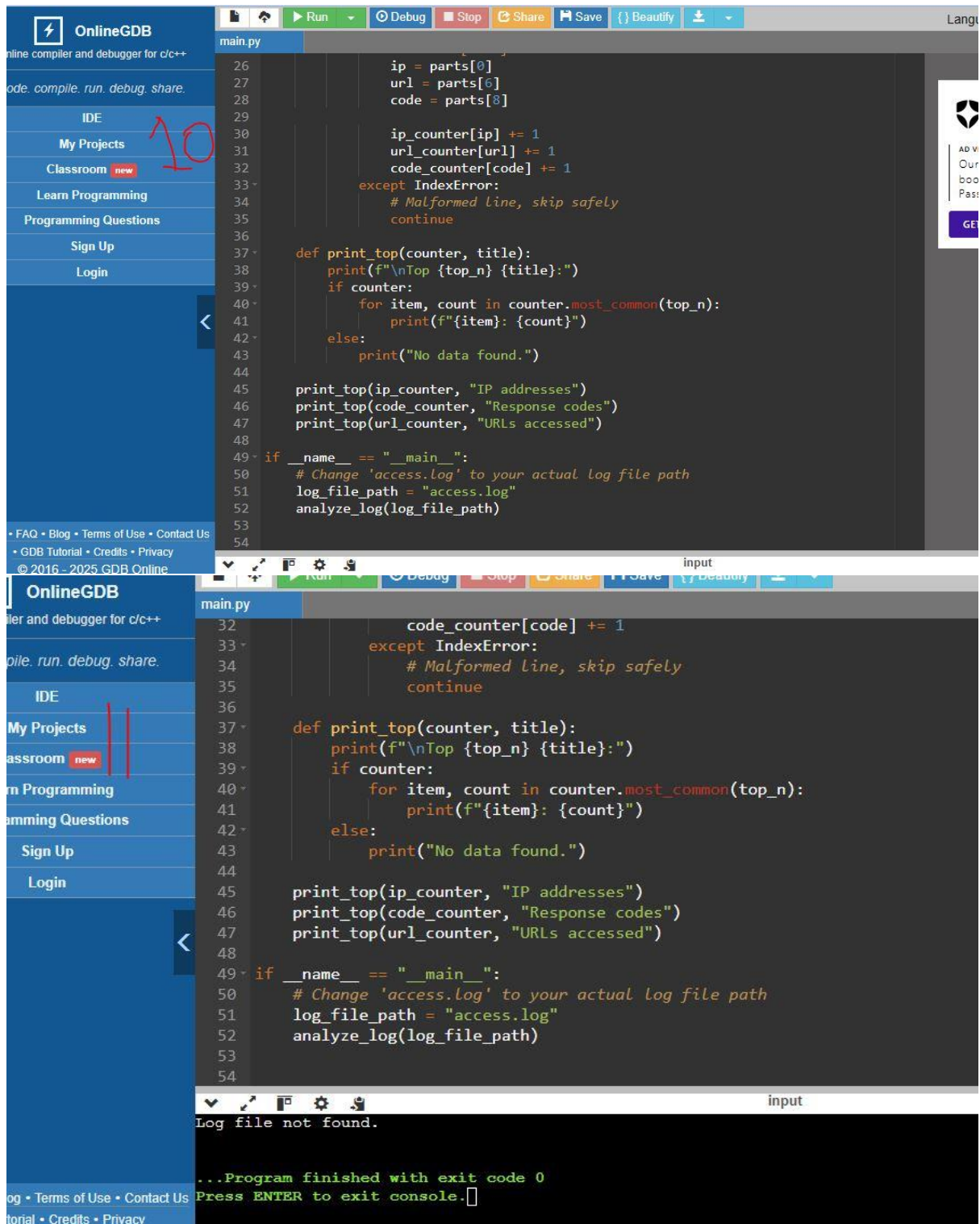
The screenshot shows the OnlineGDB IDE interface. The sidebar on the left is similar to the previous one, but with 'My Projects' circled in red. The main editor area displays a Python file named 'main.py' containing a script for analyzing log files. The code is as follows:

```
1 import os
2 from collections import Counter
3
4 def analyze_log(file_path, top_n=5):
5
6     if not os.path.isfile(file_path):
7         print("Log file not found.")
8         return
9
10    # Check file size limit (100 MB)
11    max_size = 100 * 1024 * 1024 # 100 MB
12    if os.path.getsize(file_path) > max_size:
13        print("File size exceeds 100 MB limit.")
14        return
15
16    ip_counter = Counter()
17    code_counter = Counter()
18    url_counter = Counter()
19
20    with open(file_path, 'r', encoding='utf-8', errors='ignore') as file:
21        for line in file:
22            try:
23                parts = line.split()
24                # Typical Apache/Nginx log format:
25                # IP - - [date] "METHOD URL PROTOCOL" CODE SIZE
26                ip = parts[0]
27                url = parts[6]
28                code = parts[8]
```

The console window at the bottom is currently empty. The top toolbar and sidebar navigation are consistent with the previous screenshot.



## 4. Log Analysis System:



The image displays two screenshots of the OnlineGDB IDE, showing a Python program for log analysis. The top screenshot shows the code before execution, and the bottom screenshot shows the output after execution.

**Top Screenshot (Code):**

```
main.py
26 ip = parts[0]
27 url = parts[6]
28 code = parts[8]
29
30 ip_counter[ip] += 1
31 url_counter[url] += 1
32 code_counter[code] += 1
33 except IndexError:
34     # Malformed line, skip safely
35     continue
36
37 def print_top(counter, title):
38     print(f"\nTop {top_n} {title}:")
39     if counter:
40         for item, count in counter.most_common(top_n):
41             print(f"{item}: {count}")
42     else:
43         print("No data found.")
44
45 print_top(ip_counter, "IP addresses")
46 print_top(code_counter, "Response codes")
47 print_top(url_counter, "URLs accessed")
48
49 if __name__ == "__main__":
50     # Change 'access.log' to your actual log file path
51     log_file_path = "access.log"
52     analyze_log(log_file_path)
53
54
```

**Bottom Screenshot (Output):**

```
main.py
32 code_counter[code] += 1
33 except IndexError:
34     # Malformed line, skip safely
35     continue
36
37 def print_top(counter, title):
38     print(f"\nTop {top_n} {title}:")
39     if counter:
40         for item, count in counter.most_common(top_n):
41             print(f"{item}: {count}")
42     else:
43         print("No data found.")
44
45 print_top(ip_counter, "IP addresses")
46 print_top(code_counter, "Response codes")
47 print_top(url_counter, "URLs accessed")
48
49 if __name__ == "__main__":
50     # Change 'access.log' to your actual log file path
51     log_file_path = "access.log"
52     analyze_log(log_file_path)
53
54
```

Log file not found.

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