Run | Debug | Stop | Share | Save | { } Beautify

main.py

```python
def two_sum(nums, target):
    num_to_index = {}
    for i, num in enumerate(nums):
        complement = target - num

        if complement in num_to_index:
            return [num_to_index[complement], i]

        num_to_index[num] = i

    return []
nums = [2, 7, 11, 15]
target = 9
print(two_sum(nums, target))

```

```
[0, 1]


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

Profile Performance]...

Run | Debug | Stop | Share | Save | { } Beautify

main.py

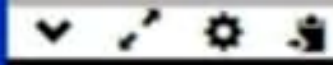
```python
class Solution:
    def longestPalindrome(self, s: str) -> str:
        def expandAroundCenter(left, right):
            while left >= 0 and right < len(s) and s[left] == s[right]:
                left -= 1
                right += 1
            return s[left + 1:right]

        if len(s) == 0:
            return ""

        longest = ""
        for i in range(len(s)):
            odd_palindrome = expandAroundCenter(i, i)
            even_palindrome = expandAroundCenter(i, i + 1)

            longest = max(longest, odd_palindrome, even_palindrome, key

        return longest


s = "babad"
solution = Solution()
print(solution.longestPalindrome(s))
```

input

```
bab


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

main.py

```python
def findMedianSortedArrays(nums1, nums2):
    nums = sorted(nums1 + nums2)
    n = len(nums)
    if n % 2 == 0:
        return (nums[n // 2 - 1] + nums[n // 2]) / 2
    else:
        return nums[n // 2]


nums1 = [1, 3]
nums2 = [2]
print(findMedianSortedArrays(nums1, nums2))
```



```
2


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

Run Debug Stop Share Save { } Beautify

main.py

```python
def length_of_longest_substring(s):
    start = maxLength = 0
    usedChars = {}

    for i in range(len(s)):
        if s[i] in usedChars and start <= usedChars[s[i]]:
            start = usedChars[s[i]] + 1
        else:
            maxLength = max(maxLength, i - start + 1)

        usedChars[s[i]] = i

    return maxLength

s = "abcabcbb"
print(length_of_longest_substring(s))
```

input

```
3


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

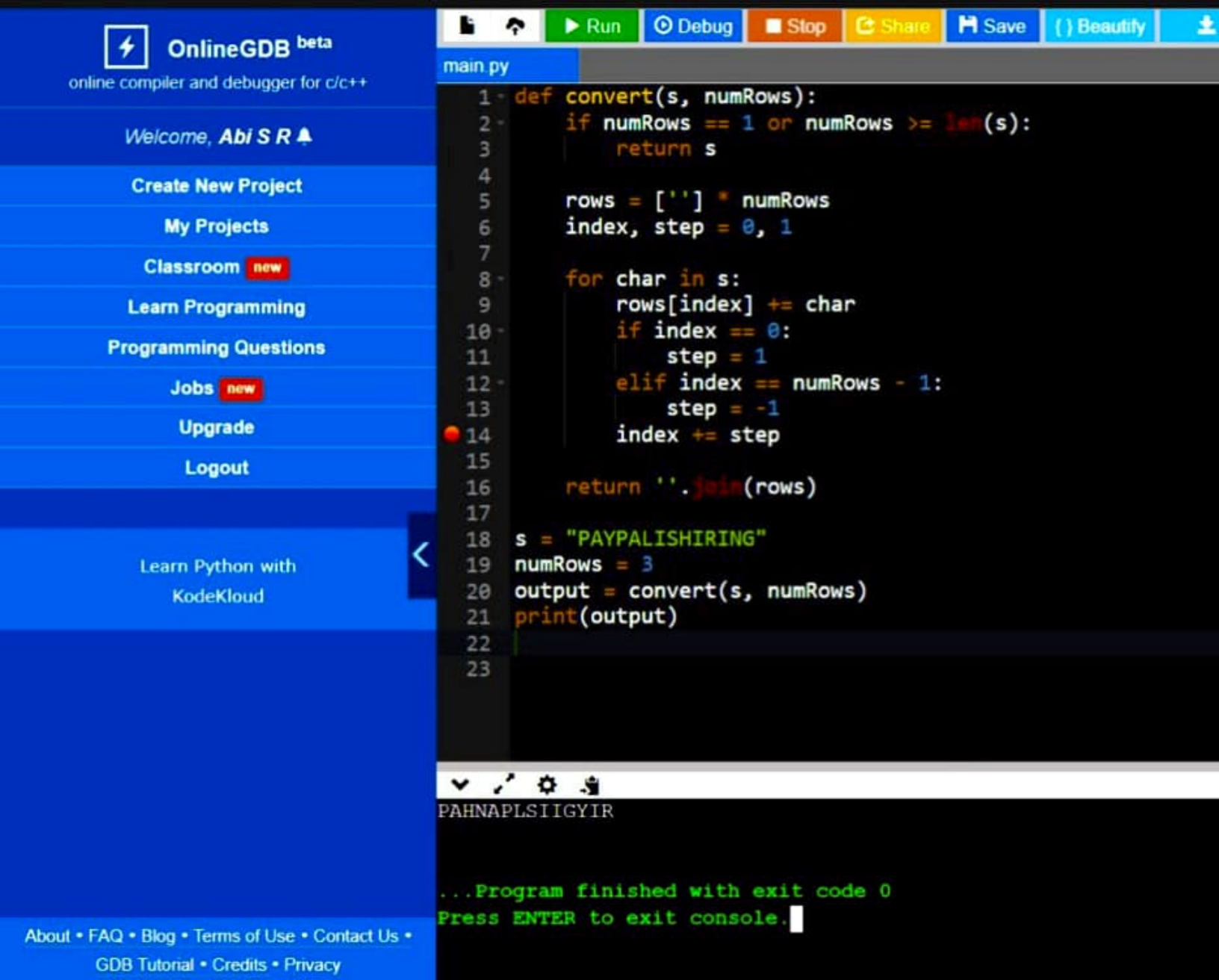
Run | Debug | Stop | Share | Save | { } Beautify

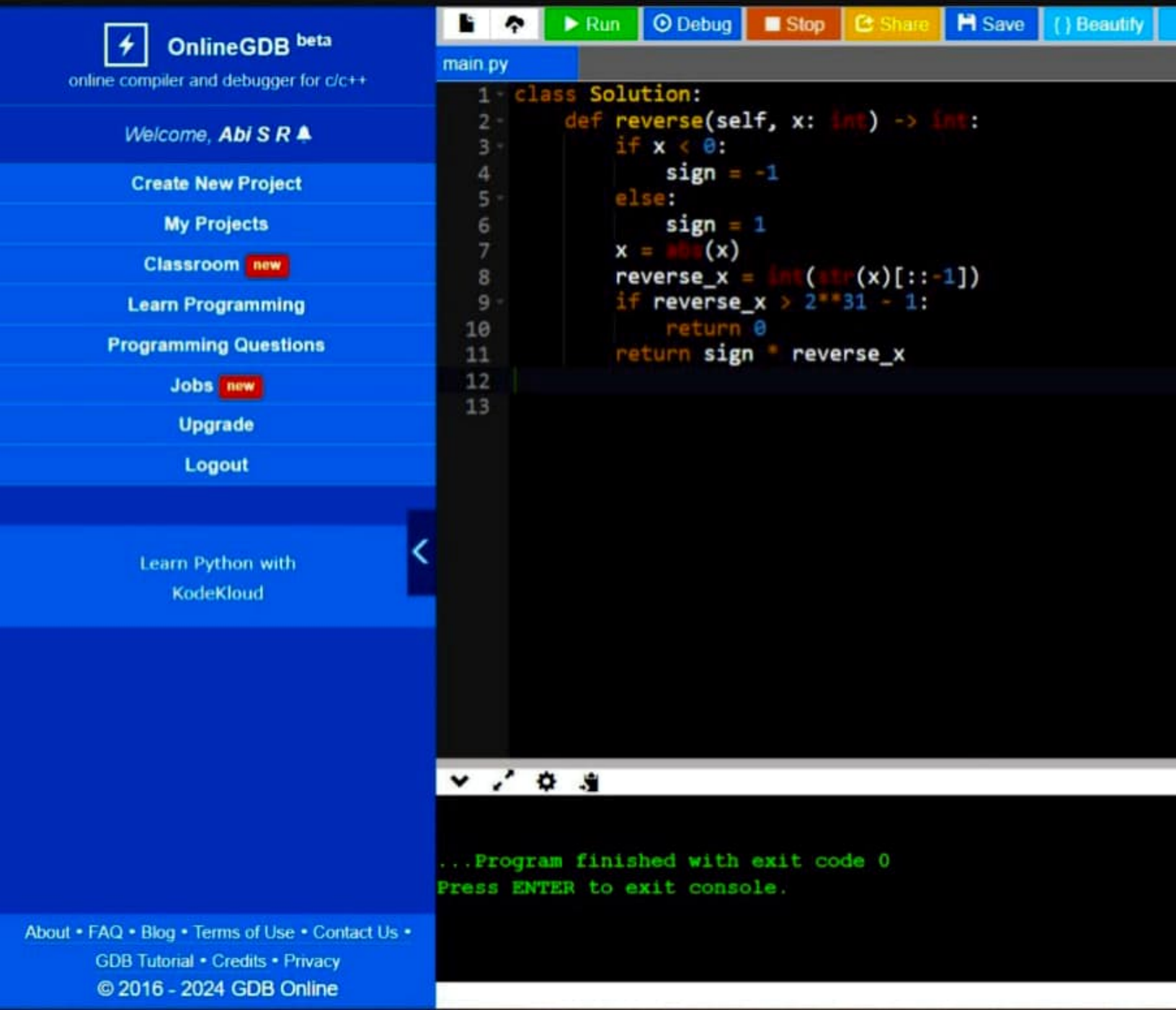
main.py

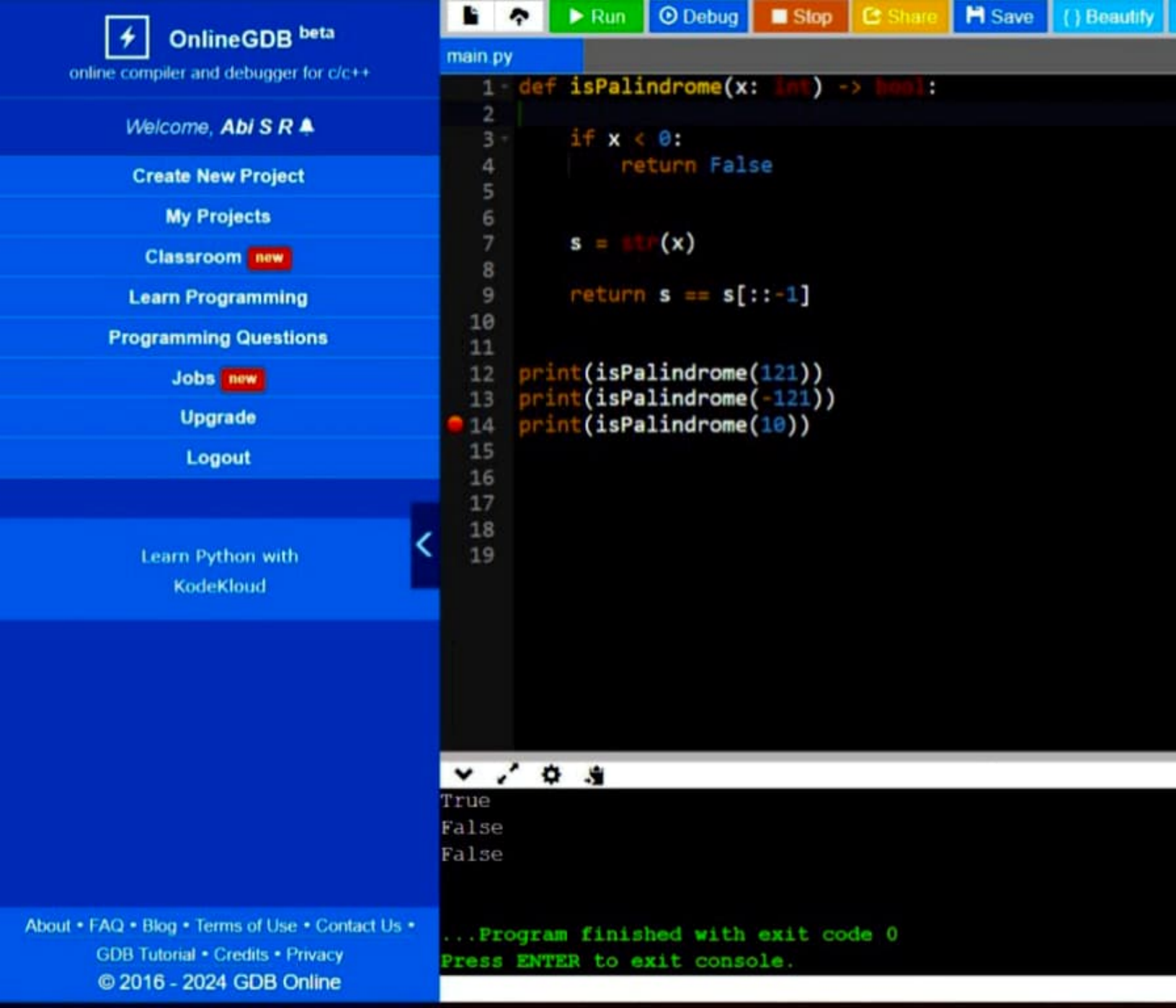
```python
class ListNode:
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

def add_two_numbers(l1, l2):
    dummy_head = ListNode(0)
    current = dummy_head
    carry = 0

    while l1 or l2 or carry:
        val1 = l1.val if l1 else 0
        val2 = l2.val if l2 else 0
        total = val1 + val2 + carry

        carry = total // 10
        current.next = ListNode(total % 10)
        current = current.next
        if l1:
            l1 = l1.next
        if l2:
            l2 = l2.next

    return dummy_head.next

def create_linked_list(values):
```

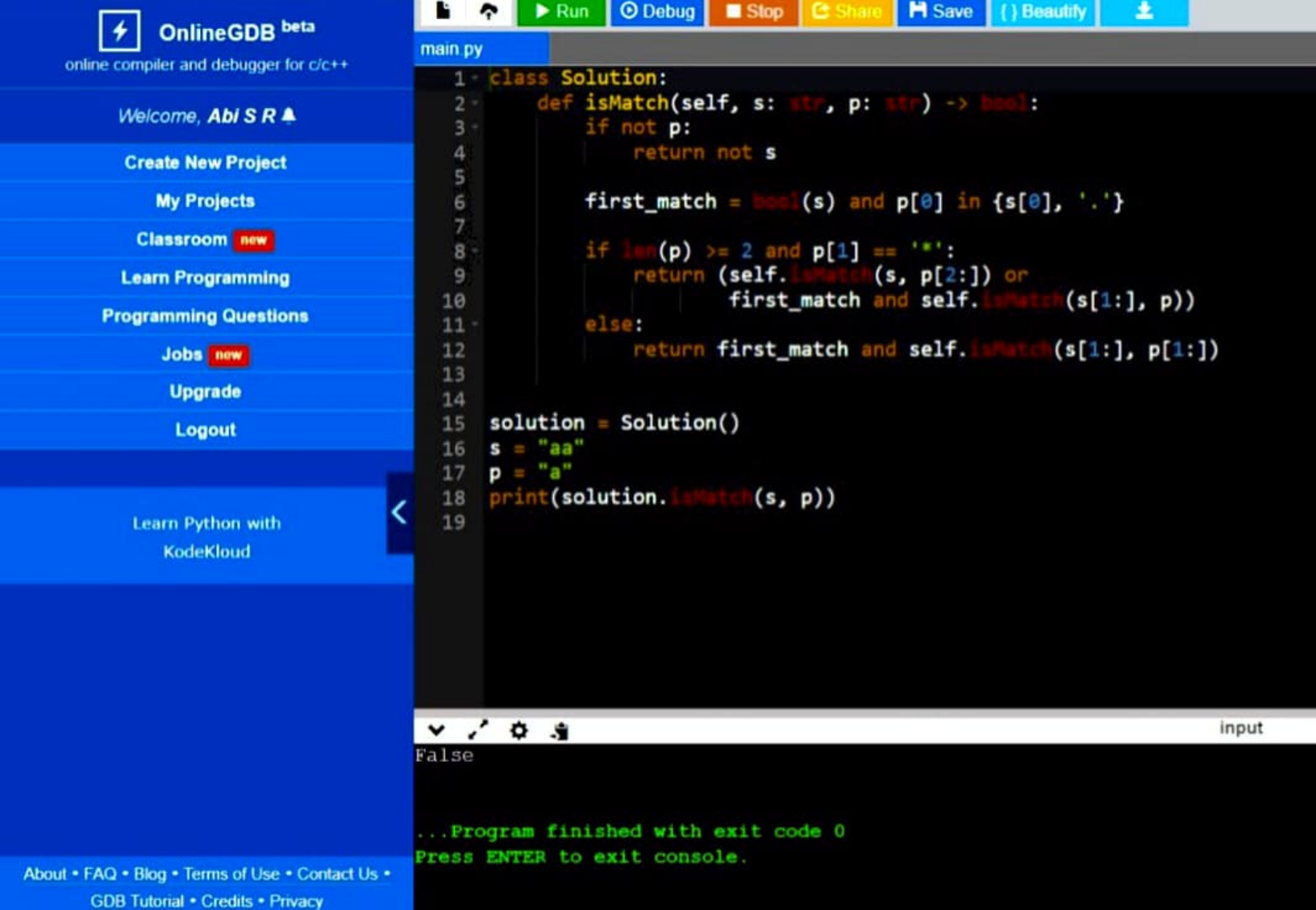
input

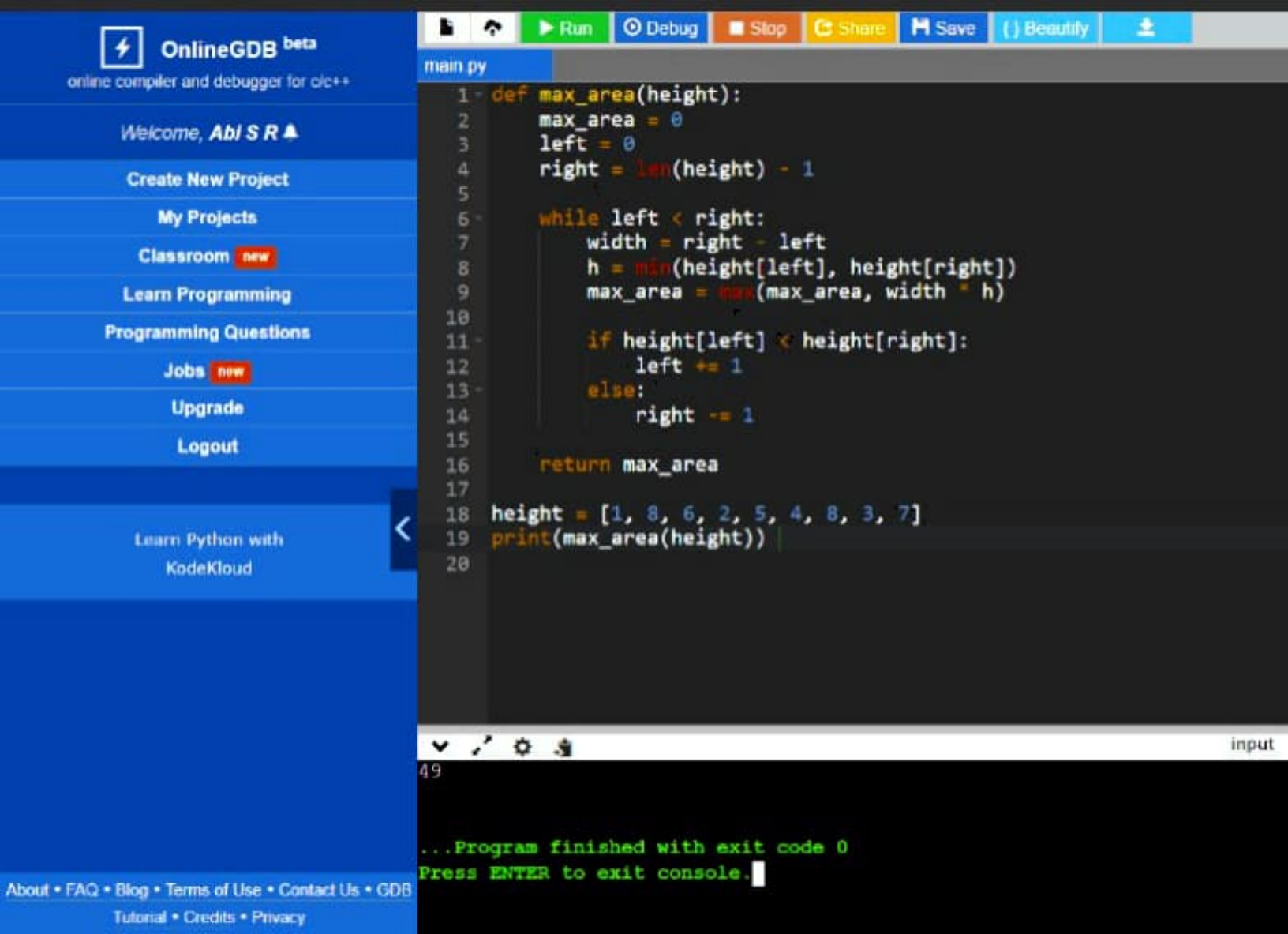
```
...Program finished with exit code 0
Press ENTER to exit console.
```


OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
Run
Debug
Stop
Share
Save
{ } Beautify
main.py
def convert(s, numRows):
if numRows == 1 or numRows >= len(s):
return s
rows = [''] * numRows
index, step = 0, 1
for char in s:
rows[index] += char
if index == 0:
step = 1
elif index == numRows - 1:
step = -1
index += step
return ''.join(rows)
s = "PAYPALISHIRING"
numRows = 3
output = convert(s, numRows)
print(output)
PAHNAPLSIIGYIR
...Program finished with exit code 0
Press ENTER to exit console.


OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
Run
Debug
Stop
Share
Save
{ } Beautify
main.py
class Solution:
def reverse(self, x: int) -> int:
if x < 0:
sign = -1
else:
sign = 1
x = abs(x)
reverse_x = int(str(x)[::-1])
if reverse_x > 2**31 - 1:
return 0
return sign * reverse_x
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online


OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
Run
Debug
Stop
Share
Save
{ } Beautify
main py
def isPalindrome(x: int) -> bool:
if x < 0:
return False
s = str(x)
return s == s[::-1]
print(isPalindrome(121))
print(isPalindrome(-121))
print(isPalindrome(10))
True
False
False
...Program finished with exit code 0
Press ENTER to exit console.
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
© 2016 - 2024 GDB Online


OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
About • FAQ • Blog • Terms of Use • Contact Us •
GDB Tutorial • Credits • Privacy
Run
Debug
Stop
Share
Save
{ } Beautify
main.py
class Solution:
def isMatch(self, s: str, p: str) -> bool:
if not p:
return not s
first_match = bool(s) and p[0] in {s[0], '.'}
if len(p) >= 2 and p[1] == '*':
return (self.isMatch(s, p[2:]) or
first_match and self.isMatch(s[1:], p))
else:
return first_match and self.isMatch(s[1:], p[1:])
solution = Solution()
s = "aa"
p = "a"
print(solution.isMatch(s, p))
input
False
...Program finished with exit code 0
Press ENTER to exit console.


OnlineGDB beta
online compiler and debugger for c/c++
Welcome, Abi S R
Create New Project
My Projects
Classroom new
Learn Programming
Programming Questions
Jobs new
Upgrade
Logout
Learn Python with
KodeKloud
About • FAQ • Blog • Terms of Use • Contact Us • GDB
Tutorial • Credits • Privacy
Run
Debug
Stop
Share
Save
{ } Beautify
main.py
def max_area(height):
max_area = 0
left = 0
right = len(height) - 1
while left < right:
width = right - left
h = min(height[left], height[right])
max_area = max(max_area, width * h)
if height[left] < height[right]:
left += 1
else:
right -= 1
return max_area
height = [1, 8, 6, 2, 5, 4, 8, 3, 7]
print(max_area(height))
input
49
...Program finished with exit code 0
Press ENTER to exit console.