untitled

File Edit Format Run Options Window Help

```
#code for subarray with given sum
def find_subarray(arr, S):
for i in range(len(arr)):
sum = 0
for j in range(i, len(arr)):
sum += arr[j]
if sum == S:
return [i, j]
return -1

arr = [1, 2, 3, 4, 5]
S = 9
print(find_subarray(arr, S))
```

Ln: 13 Col: 26

untitled

File Edit Format Run Options Window Help

```
#code for binary search
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

arr = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]
target = 23
print(binary_search(arr, target))
```

Ln: 16 Col: 33

untitled

File Edit Format Run Options Window Help

```
#code for bubble sort
def bubble_sort(lst):
n = len(lst)
for i in range(n-1):
for j in range(n-i-1):
if lst[j] > lst[j+1]:
lst[j], lst[j+1] = lst[j+1], lst[j]
return lst

numbers = [64, 34, 25, 12, 22, 11, 90]
print("Original List:", numbers)
print("Sorted List:", bubble_sort(numbers))
```

Ln: 2 Col: 0

untitled

File Edit Format Run Options Window Help

```
#code for pythagorean triplet
def is_pythagorean_triplet(a, b, c):
return a**2 + b**2 == c**2 or a**2 + c**2 == b**2 or b**2 + c**2 == a**2

a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))

print("Is Pythagorean Triplet:", is_pythagorean_triplet(a, b, c))
```

Ln: 1 Col: 29

untitled

File Edit Format Run Options Window Help

```
#code for count words in a sentence
def word_count(sentence):
return len(sentence.split())

sentence = input("Enter a sentence: ")
print("Word Count:", word_count(sentence))
```

Ln: 1 Col: 35

untitled

File Edit Format Run Options Window Help

```
#code for longest word in a sentence
def longest_word(sentence):
words = sentence.split()
longest = max(words, key=len)
return longest

sentence = input("Enter a sentence: ")
print("Longest Word:", longest_word(sentence))
```

Ln: 1 Col: 36

untitled

File Edit Format Run Options Window Help

```
#code for balanced parentheses
def is_balanced(s):
    stack = []
    parentheses_map = {")": "(", "}": "{", "]": "["}

    for char in s:
        if char in parentheses_map.values():
            stack.append(char)
        elif char in parentheses_map.keys():
            if not stack or parentheses_map[char] != stack.pop():
                return False

    return not stack

s = input("Enter a string of parentheses: ")
print("Is Balanced:", is_balanced(s))
```

Ln: 1 Col: 31

untitled

File Edit Format Run Options Window Help

```
#code for missing number
def find_missing_number(lst):
    n = len(lst) + 1
    total_sum = n * (n + 1) // 2
    actual_sum = sum(lst)
    return total_sum - actual_sum

lst = [1, 2, 3, 5]
print("Missing Number:", find_missing_number(lst))
```

Ln: 1 Col: 24