

Problem Statement:

Find the First Non-Repeating Character

Write a program to find the first non-repeating character in a string. For input "swiss", the output should be "w". You cannot use any built-in string or character frequency counting functions.

Instructions: Implement manual string traversal and counting logic to solve the problem.

Coding in ruby:

```
def nrc(str) # this is a function to check non repeating character the given string

  c_count = {} # here we store the character count using hash, collection of key value pairs

  str.each_char do |char| #iterate each character in a string

    if c_count.key?(char)

      c_count[char] += 1 #if the character char is already a key in the c_count hash, its value is
      incremented by 1

    else

      c_count[char] = 1 #if the character is not yet a key in the hash, it is added with a value of 1
    end
  end

  #c_count will store the count of occurrences of each character in the string.

  str.each_char do |char| #another iteration over each character of the string is performed.

    return char if c_count[char] == 1 # it checks if the count of the character char in c_count is 1. If
    it is, that means the character is non-repeating.

  end

  nil #if there are no non-repeating characters, the method returns nil
end

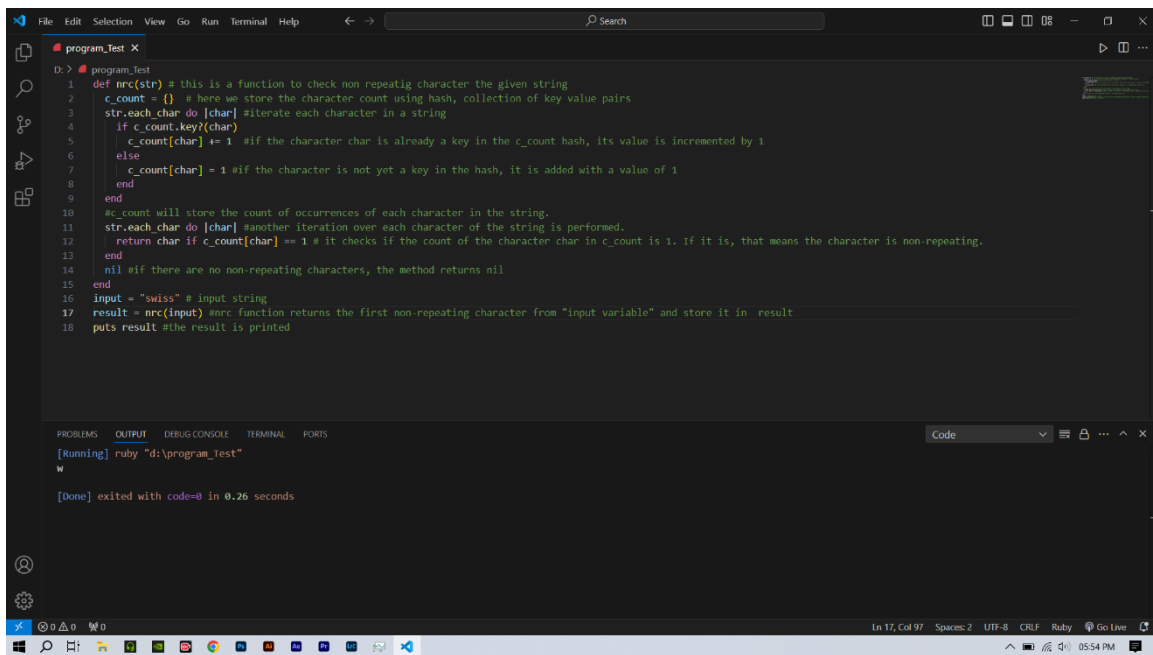
input = "swiss" # input string (change this string to for alter results like google,hash agile, ect...)

result = nrc(input) #nrc function returns the first non-repeating character from "input variable"
and store it in result

puts result #the result is printed
```

Screenshots:

1. Input: swiss output: w



```
1 def nrc(str) # this is a function to check non repeating character the given string
2   c_count = {} # here we store the character count using hash, collection of key value pairs
3   str.each_char do |char| #iterate each character in a string
4     if c_count.key?(char)
5       c_count[char] += 1 #if the character char is already a key in the c_count hash, its value is incremented by 1
6     else
7       c_count[char] = 1 #if the character is not yet a key in the hash, it is added with a value of 1
8     end
9   end
10  #c_count will store the count of occurrences of each character in the string.
11  str.each_char do |char| #another iteration over each character of the string is performed.
12    return char if c_count[char] == 1 # it checks if the count of the character char in c_count is 1. If it is, that means the character is non-repeating.
13  end
14  nil #if there are no non-repeating characters, the method returns nil
15 end
16 input = "swiss" # input string
17 result = nrc(input) #nrc function returns the first non-repeating character from "input variable" and store it in result
18 puts result #the result is printed
```

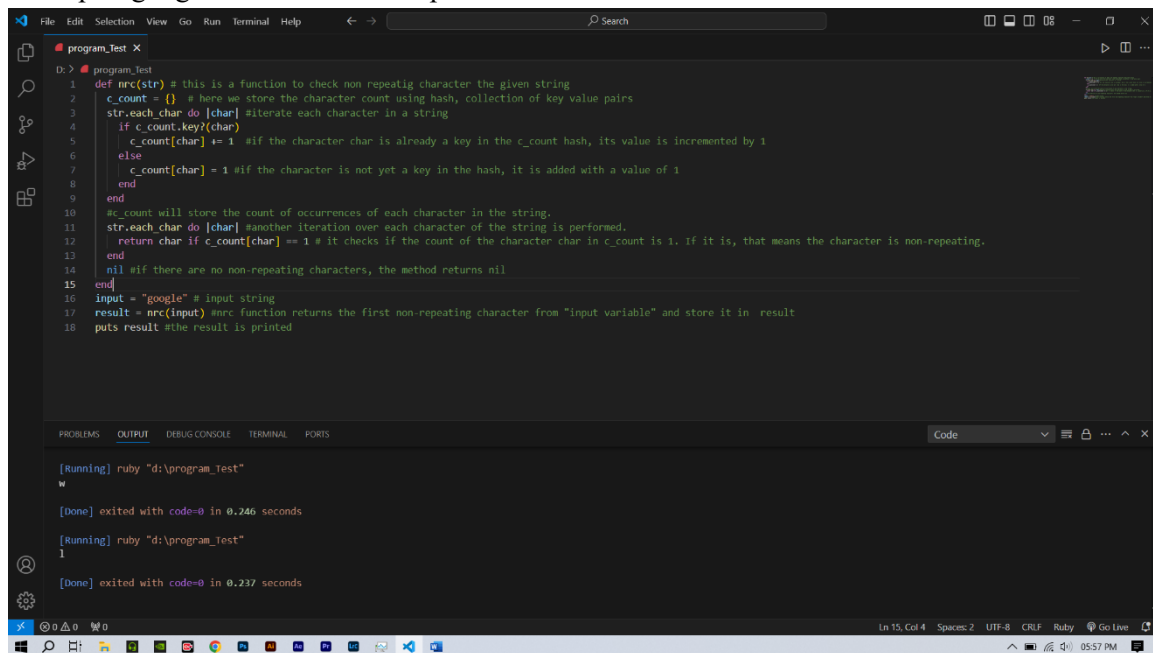
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[Running] ruby "d:\program_test"

w

[Done] exited with code=0 in 0.26 seconds

2. Input: google output: l



```
1 def nrc(str) # this is a function to check non repeating character the given string
2   c_count = {} # here we store the character count using hash, collection of key value pairs
3   str.each_char do |char| #iterate each character in a string
4     if c_count.key?(char)
5       c_count[char] += 1 #if the character char is already a key in the c_count hash, its value is incremented by 1
6     else
7       c_count[char] = 1 #if the character is not yet a key in the hash, it is added with a value of 1
8     end
9   end
10  #c_count will store the count of occurrences of each character in the string.
11  str.each_char do |char| #another iteration over each character of the string is performed.
12    return char if c_count[char] == 1 # it checks if the count of the character char in c_count is 1. If it is, that means the character is non-repeating.
13  end
14  nil #if there are no non-repeating characters, the method returns nil
15 end
16 input = "google" # input string
17 result = nrc(input) #nrc function returns the first non-repeating character from "input variable" and store it in result
18 puts result #the result is printed
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

[Running] ruby "d:\program_test"

w

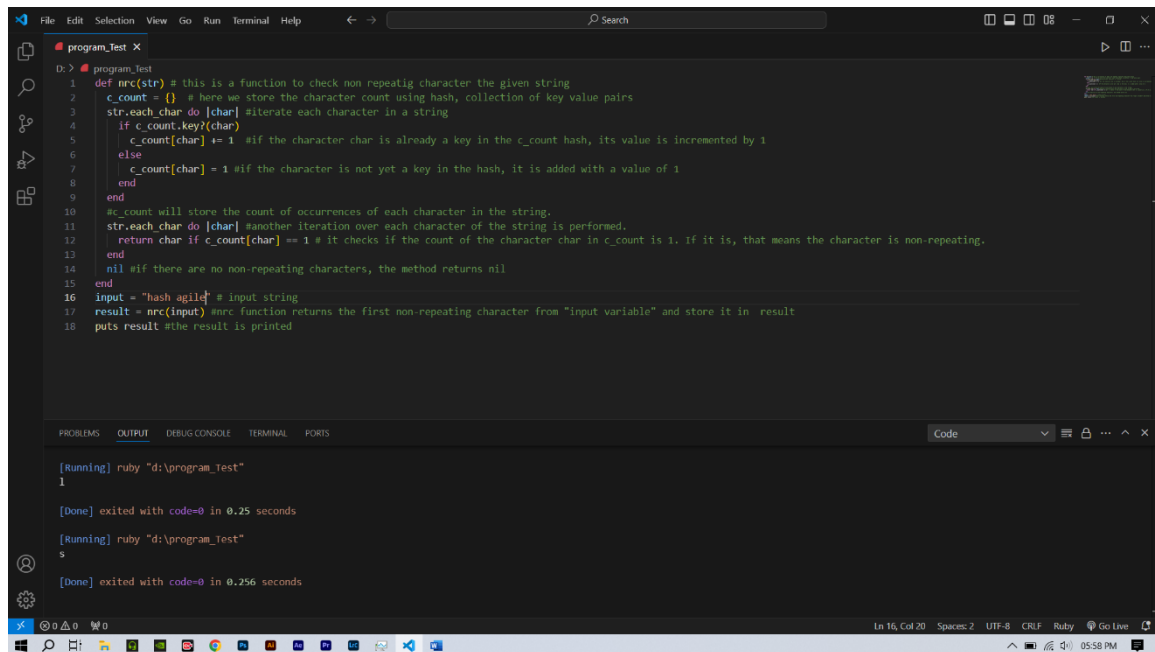
[Done] exited with code=0 in 0.246 seconds

[Running] ruby "d:\program_test"

l

[Done] exited with code=0 in 0.237 seconds

3. Input: hash agile output: h



The screenshot shows a Ruby IDE with a file named `program_test`. The code defines a method `nrc(str)` that finds the first non-repeating character in a string. It uses a hash `c_count` to track the frequency of each character. The input string is `"hash agile"`, and the output is `h`.

```
1 def nrc(str) # this is a function to check non repeating character the given string
2   c_count = {} # here we store the character count using hash, collection of key value pairs
3   str.each_char do |char| #iterate each character in a string
4     if c_count.key?(char)
5       c_count[char] += 1 #if the character char is already a key in the c_count hash, its value is incremented by 1
6     else
7       c_count[char] = 1 #if the character is not yet a key in the hash, it is added with a value of 1
8     end
9   end
10  #c_count will store the count of occurrences of each character in the string.
11  str.each_char do |char| #another iteration over each character of the string is performed.
12    return char if c_count[char] == 1 # it checks if the count of the character char in c_count is 1. If it is, that means the character is non-repeating.
13  end
14  nil #if there are no non-repeating characters, the method returns nil
15 end
16 input = "hash agile" # input string
17 result = nrc(input) #nrc function returns the first non-repeating character from "input variable" and store it in result
18 puts result #the result is printed
```

The output console shows the following results:

```
[Running] ruby "d:\program_test"
1
[Done] exited with code=0 in 0.25 seconds
[Running] ruby "d:\program_test"
h
[Done] exited with code=0 in 0.256 seconds
```

Sample outputs:

- | | |
|----------------------|-----------|
| 1. Input: swiss | output: w |
| 2. Input: google | output: l |
| 3. Input: hash agile | output: h |