

Laboratory 2
Data Structures

Santiago Cardona 160004910
Juan Aristizabal 160004903

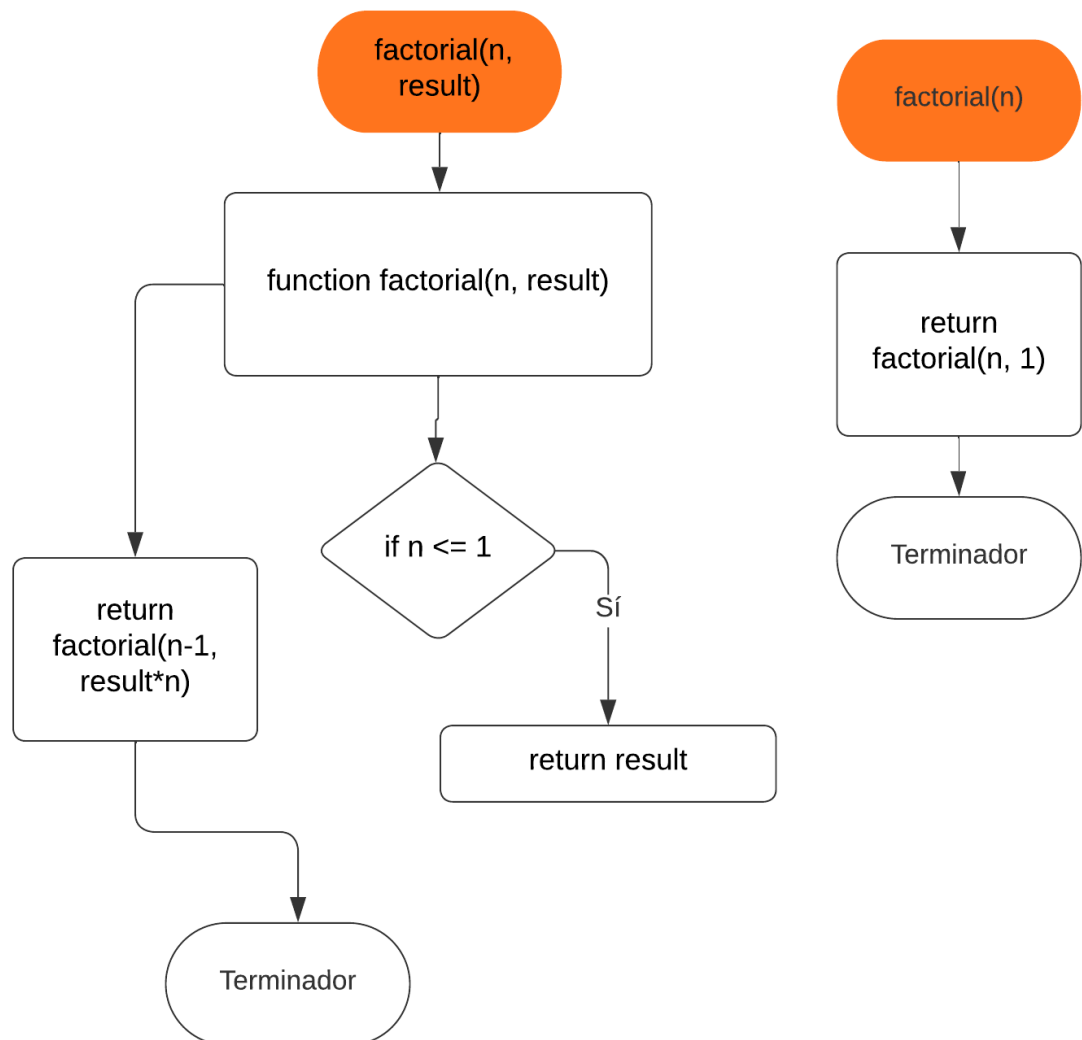
September 07, 2023

1. Program 1

1.1 Statement

Write a tail version of the factorial function and add a non recursive function to call it. Test the function in a program.

1.2 Flowchart



1.3 Pseudocode

```
function factorial(n, result)
  if n <= 1
    return result
```

```

        end if
        return factorial(n-1, result*n)
    end function

function factorial(n)
    return factorial(n, 1)
end function

function main()
    print "Factorial de 5 " + factorial(5)
    print "Factorial de 4 " + factorial(4)
end function

```

1.4 Source code

```

int factorial (int n,int result ){
    if (n <= 1){
        return result;
    }
    return factorial(n-1,result*n);
}

int factorial (int n){
    return factorial (n,1);
}

int main(){
    cout<<"Factorial de 5 "<<factorial(5)<<endl;
    cout<<"Factorial de 4 "<<factorial(4)<<endl;
    return 0;
}

```

1.5 Description

This function receives two parameters: an integer n and an integer result. The function is divided into two cases:

- Base case: if n is less than or equal to 1, the value of result is returned.
- General case: The function is called recursively with n decremented by 1 and the value of result multiplied by n. The result of the recursive call is returned.
- Helper function: calls the factorial function

1.6 Results

```

10  }
11
12      return factorial(n-1, result*n);
13  }
14  int factorial(int n){
15      return factorial(n, 1);
16  }
17  int main(){
18      cout<<"Factorial de 5 "<<factorial(5)<<endl;
19      cout<<"Factorial de 4 "<<factorial(4)<<endl;
20
21      return 0;
22  }

```

Run: ejercicio_1.cpp

```

Factorial de 5 120
Factorial de 4 24

```

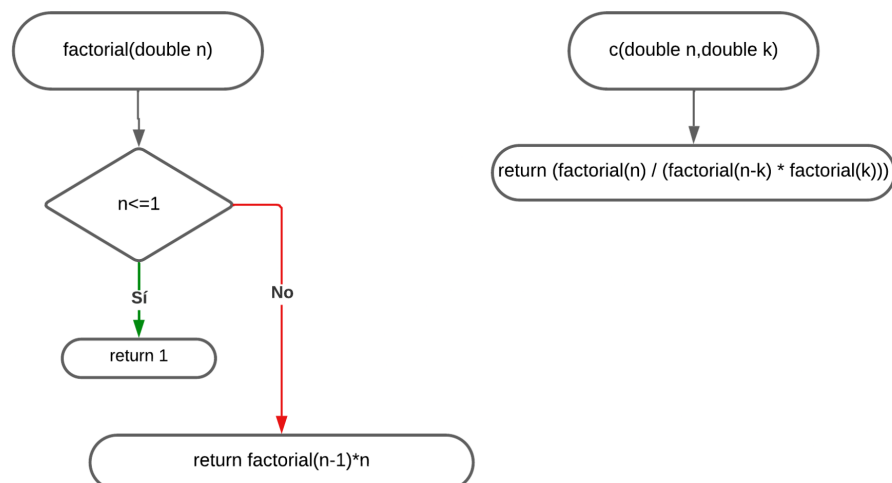
Process finished with exit code 0

2. Program 3

2.1 Statement

Write a recursive function to find the combination of n objects k at a time. Then write a program to test it. The formula is given below: $C(n, k) = \frac{\text{factorial}(n)}{(\text{factorial}(n - k) * \text{factorial}(k))}$

2.2 Flowchart



2.3 Pseudocode

2.4 Source code

```

double factorial(double n){
    if (n<=1) return 1;
    return factorial(n-1)*n;
}

double c(double n,double k){
    return (factorial(n) / (factorial(n-k) * factorial(k)));
}

int main() {
    double n,k;
    n = 5;
    k = 3;
    cout << "Las combinaciones posibles de c(" << n
        << "," << k << ") son: " << c(n,k) << endl;
    k = 4;
    cout << "Las combinaciones posibles de c(" << n
        << "," << k << ") son: " << c(n,k) << endl;
    n = 20;
    k = 3;
    cout << "Las combinaciones posibles de c(" << n
        << "," << k << ") son: " << c(n,k) << endl;

    return 0;
}

```

2.5 Description

A function that calculates the number of combinations by performing a mathematical operation is performed. In order to perform this operation, it makes use of a recursive function with which it calculates the factorial of a number.

2.6 Results

```

19 int main() {
20     double n,k;
21     n = 5;
22     k = 3;
23     cout << "Las combinaciones posibles de c(" << n << "," << k << ") son: " << c(n,k) << endl;
24     k = 4;
25     cout << "Las combinaciones posibles de c(" << n << "," << k << ") son: " << c(n,k) << endl;
26     n = 20;

```

Run

```

Las combinaciones posibles de c(5,3) son: 10
Las combinaciones posibles de c(5,4) son: 5
Las combinaciones posibles de c(20,3) son: 1140
Process finished with exit code 0

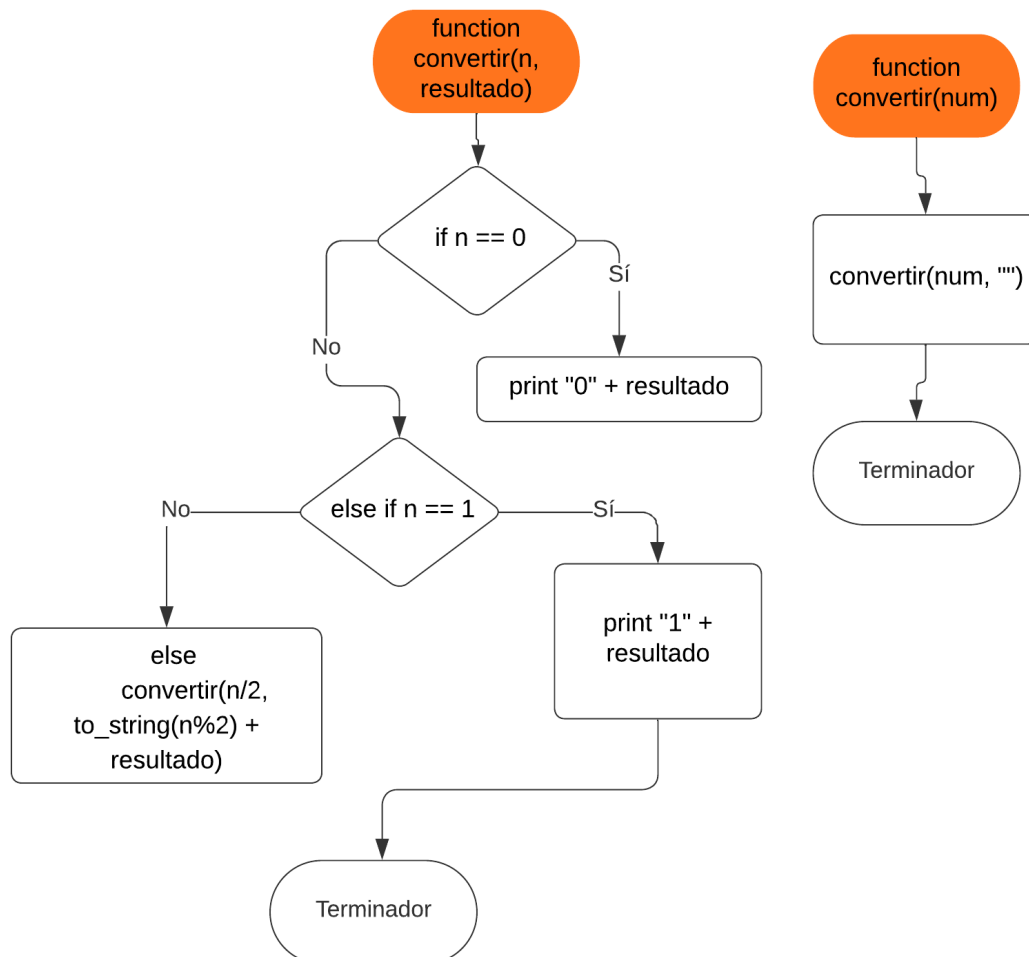
```

3. Program 5

3.1 Statement

Write a tail recursive function to convert a decimal integer to a binary string and then use a non recursive function to call it. For example, the decimal 78 will be changed to "1001110". Test your functions in a program.

3.2 Flowchart



3.3 Pseudocode

```

function convertir(n, resultado)
  if n == 0
    print "0" + resultado
  else if n == 1
    print "1" + resultado
  else
    convertir(n/2, to_string(n%2) + resultado)
  end if
end function
  
```

```

function convertir(num)
  convertir(num, "")
end function
  
```

```

function main()
  
```

```
    print "4: "  
    convertir(120)  
end function
```

3.4 Source code

```
void convertir(int n,string resultado){  
    if(n==0) cout<<"0"+resultado;  
    else if(n==1) cout<<"1"+resultado;  
    else convertir (n/2, to_string(n%2)+ resultado);  
}  
void convertir(int num){convertir(num,"");}  
int main(){  
    cout<<"4: "  
    convertir(4);  
    return 0;  
}
```

3.5 Description

Recursive function with queue to convert an integer to binary

This function receives two parameters: an integer n and a result string to store the binary digits. The function is divided into three cases:

- Base case 1: If n is equal to 0, a 0 is added to the beginning of the result string and the string is returned.
- Base case 2: If n is equal to 1, a 1 is added to the beginning of the result string and the string is returned.
- General case: The function is called recursively with n divided by 2 as the first parameter and the modulo of n&2 as the second parameter. The result of the recursive call is concatenated with the result string and the string is returned.
- Helper function: Makes the call to the function convert.

3.6 Results

```

ejercicio_1.exe
ejercicio_2.cpp
ejercicio_2.exe
ejercicio_3.cpp
External Libraries
Scratches and Consoles

3  * */
4  #include<iostream>
5  using namespace std;
6  void convertir(int n,string resultado){
7      if(n==0) cout<<"0"+resultado;
8      else if(n==1) cout<<"1"+resultado;
9      else convertir ( n: n/2, resultado: to_string( val: n%2)+ resultado);
10 }
11 void convertir(int num){convertir( n: num, resultado: "");}
12 i main(){
13     cout<<"78: "; convertir( num: 78);
14     return 0;
15 }

f main

un: ejercicio_2.cpp x
78: 1001110
Process finished with exit code 0

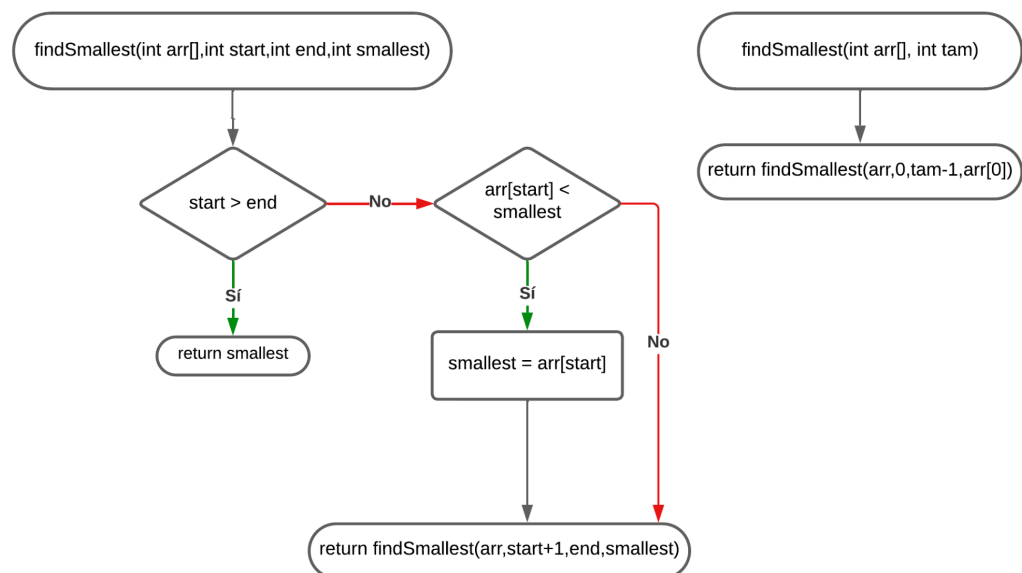
```

4. Program 7

4.1 Statement

Write a tail recursive function to find the smallest integer in an array of integers. Then use a non recursive function to call it. Test your function in a program with at least three arrays of 10 integers. Note that when we look for the smallest element in an array, we must keep track of the smallest element and the next index to check. This means that your helper function must have two extra parameters.

4.2 Flowchart



4.3 Pseudocode

Function `findSmallest(arr[],start,end,smallest)`

if "start" is greater than "end", do

return the variable "smallest"

else if the position "start" of the array "arr" is less than "smallest", do

```
        set the value of "smallest" as the value of the position "start" of the array
    return findSmallest(arr,start+1,end,smallest)
end function
```

```
Function findSmallest(arr[],tam)
    return findSmallest(arr,0,tam-1,arr[0])
end function
```

4.4 Source code

```
int findSmallest(int arr[],int start,int end,int
smallest){
    if (start > end) return smallest;
    else if (arr[start] < smallest) smallest = arr[start];
    return findSmallest(arr,start+1,end,smallest);
}

int findSmallest(int arr[], int tam){
    return findSmallest(arr,0,tam-1,arr[0]);
}

int main() {
    int arr1[] = {9,8,4,5,6,7, 36, 84, 27, 2};
    int arr2[] = {73,45,48,5,8,9, 7, 30, 1, 33};
    int arr3[] = {1,-1,344,5,1,2, -20, 0, 12, 16};
    cout << findSmallest(arr1,10) << endl;
    cout << findSmallest(arr2,10) << endl;
    cout << findSmallest(arr3,10) << endl;

    return 0;
}
```

4.5 Description

A recursive helper function is created that receives 4 data, this function has a base case in which it asks if the variable "start", which represents the position in which it starts, is greater than the position in which it ends, then it returns the smallest variable; it also has another conditional, which asks if the element in the element in the "start" position of the array is smaller than the "smallest" variable, which stores the number of the array that is smaller so far; if so, the "smallest" variable takes the value of the element that is in the "start" position of the array. After that, the recursive call is made, but now the same variables are sent to the function, only now a 1 is added to the variable "start". Then, this function is called from the main function, sending the necessary parameters.

4.6 Results


```

13
14 int main() {
15     int arr1[] = { [0]: 9, [1]: 8, [2]: 4, [3]: 5, [4]: 6, [5]: 7, [6]: 36, [7]: 84, [8]: 27, [9]: 2 };
16     int arr2[] = { [0]: 73, [1]: 45, [2]: 48, [3]: 5, [4]: 8, [5]: 9, [6]: 7, [7]: 30, [8]: 1, [9]: 33 };
17     int arr3[] = { [0]: 1, [1]: -1, [2]: 344, [3]: 5, [4]: 1, [5]: 2, [6]: -20, [7]: 0, [8]: 12, [9]: 16 };
18     cout << findSmallest(arr, arr1, tam: 10) << endl;
19     cout << findSmallest(arr, arr2, tam: 10) << endl;
20     cout << findSmallest(arr, arr3, tam: 10) << endl;
21
22     return 0;
23 }

```

Run P7 x

"C:\Users\Juan\JArizabal\U\TrabajosU\Estructuras de datos\L2\P7\cmake-build-debug\P7.exe"

2
1
-20

P7 > main.cpp clang-tidy LF UTF-8 4 spaces C++ P7 Debug

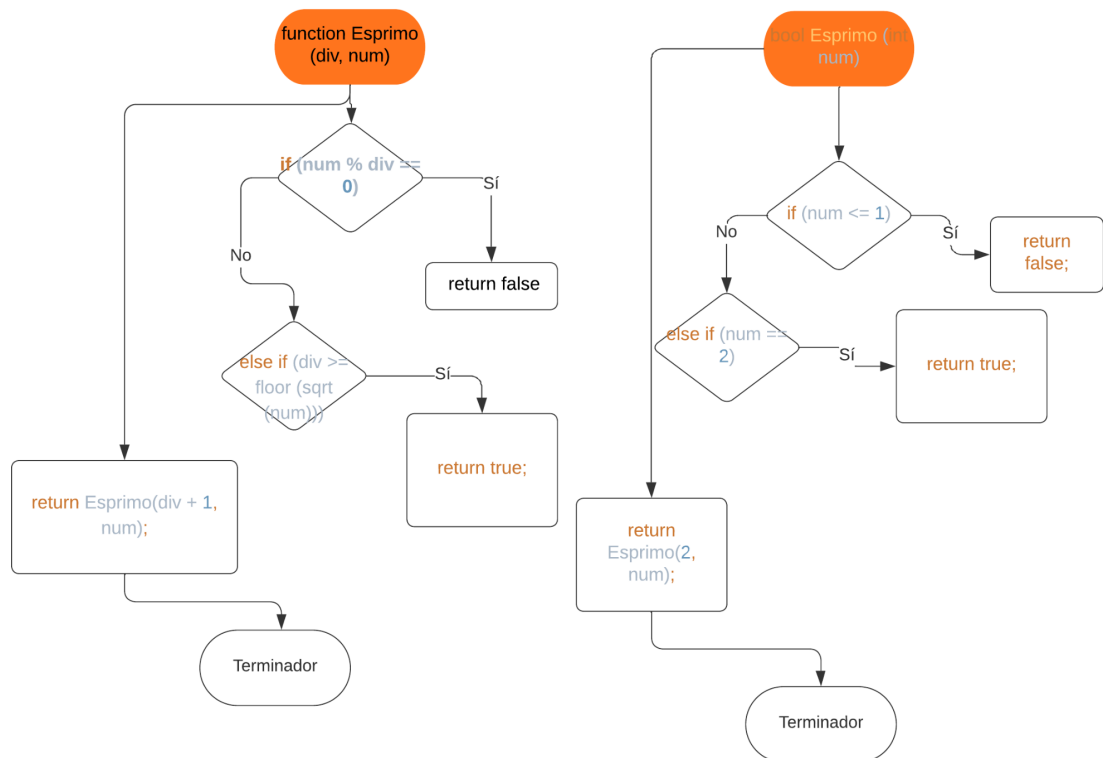
5. Program 9

5.1 Statement

Write a tail recursive function and the corresponding nonrecursive function to test if an integer is prime. Remember that a prime number is only divisible by 1 and itself, but we can check the numbers from 1 to the square root of that number to be sure that it is prime. Remember that 1 is not a prime.

Mathematicians divide the positive integers into three groups: composite, prime, and 1.

5.2 Flowchart



5.3 Pseudocode

```

function Esprimo (div, num)
    if num modulo div is equal to 0 then
        return false
    else if div is greater than or equal to the square root of num rounded down
then
        return true
    else
        return Esprimo(div + 1, num)
    end if
end function

function Esprimo (num)
    if num is less than or equal to 1 then
        return false
    else if num is equal to 2 then
        return true
    else
        return Esprimo(2, num)
    end if
end function

function main ()
    write "Is 13 a prime number? ", Esprimo(13)
end function main

```

5.4 Source code

```

bool Esprimo (int div, int num){
if (num % div == 0)
{
return false;
}
else if (div >= floor (sqrt (num)))
{
return true;
}
return Esprimo(div + 1, num);
}

bool Esprimo (int num){
if (num <= 1)
{
return false;
}
else if (num == 2)
{
return true;
}
return Esprimo(2, num);
}

```

```

int main ( )
{
    cout << "13 es primo? " << boolalpha << Esprimo(13) << endl;
    return 0;
}

```

5.5 Description

The function `Esprimo` is defined with two parameters: `div` and `num`. The `div` parameter is the divisor and the `num` parameter is the number to be tested.

The function first checks if the number is divisible by the divisor. If it is, then the function returns false.

Otherwise, the function checks if the divisor is greater than or equal to the square root of the number. If it is, then the function returns true.

Otherwise, the function recursively calls itself with the divisor plus 1 as the new divisor.

The main function of the code simply calls the `Esprimo` function with the number 13 as the argument.

5.6 Results

The screenshot shows a C++ IDE with a project named 'LAB 2'. The file explorer on the left lists files: `ejercicio_1.cpp`, `ejercicio_1.exe`, `ejercicio_2.cpp`, `ejercicio_2.exe`, `ejercicio_3.cpp`, and `ejercicio_3.exe`. The main editor displays the code for `ejercicio_3.cpp`, which includes the `Esprimo` function and the `main` function. The `Esprimo` function is defined as follows:

```

26 {return true;
27 }
28 return Esprimo( div+2, num);
29 }
30
31 int main ( )
32 {
33     cout << "13 es primo? " << boolalpha << Esprimo( num: 13) << endl;
34     return 0;

```

The console output at the bottom shows the execution of `ejercicio_3.exe` with the following output:

```

"C:\Users\santi\OneDrive\Documents\Unillanos\Tercer semestre\Estructuras\LAB 2\ejercicio_3.exe"
13 es primo? true
Process finished with exit code 0

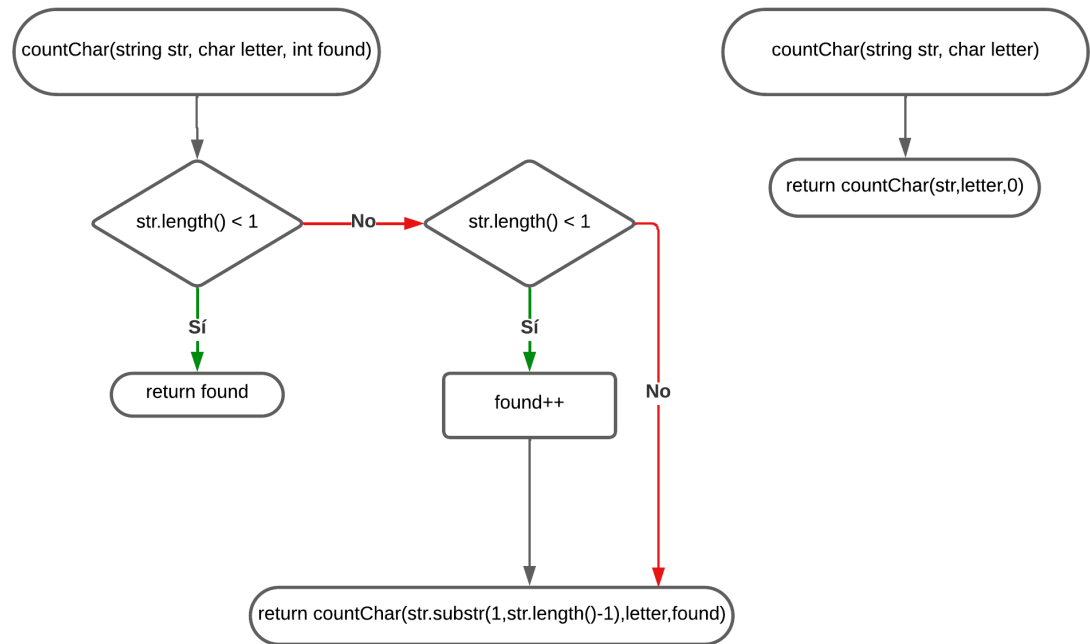
```

6. Program 11

6.1 Statement

Write a recursive function that counts and prints the number of a given character in a string. Test the function in a program.

6.2 Flowchart



6.3 Pseudocode

Function countChar(str,letter,found)
 if string length is less than 1, do
 return the variable "found"
 else if the position zero of the string "str" is equal than "letter", do
 increment the variable "found"
 return countChar(str.substr(1,str.length()-1),letter,found)
 end function

Function countChar(str,letter)
 return countChar(str,letter,0)
 end function

6.4 Source code

```

int countChar(string str, char letter, int found){
    if (str.length() < 1) return found;
    else if (str[0] == letter) found++;
    return countChar(str.substr(1, str.length()-1), letter, found);
}

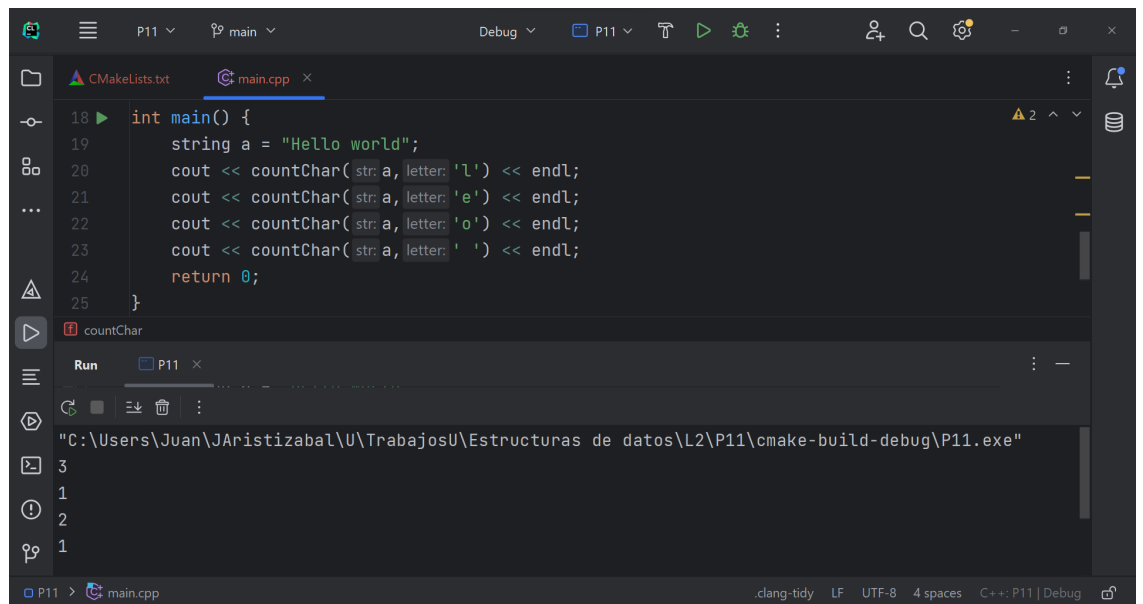
int countChar(string str, char letter){
    return countChar(str, letter, 0);
}

int main() {
    string a = "Hello world";
    cout << countChar(a, 'l') << endl;
    cout << countChar(a, 'e') << endl;
    cout << countChar(a, 'o') << endl;
    cout << countChar(a, ' ') << endl;
    return 0;
}
  
```

6.5 Description

A recursive function is elaborated, with the base case when the size of the string is less than 1, in addition, it uses another conditional to determine if the first character of the string is equal to the searched character. After that, the recursive call is made, to which a substring is sent without the first position it had, the character to look for is sent and the variable "found" that stores the number of times that the searched character appears in the string. Then, this helper function is called, from another function which would be the main function.

6.6 Results



The screenshot displays a C++ IDE with a dark theme. The main editor window shows the following code in `main.cpp`:

```
18 int main() {  
19     string a = "Hello world";  
20     cout << countChar(str: a, letter: 'l') << endl;  
21     cout << countChar(str: a, letter: 'e') << endl;  
22     cout << countChar(str: a, letter: 'o') << endl;  
23     cout << countChar(str: a, letter: ' ') << endl;  
24     return 0;  
25 }
```

Below the editor, the `Run` panel shows the execution of the program. The output is as follows:

```
"C:\Users\Juan\JAriztizabal\U\TrabajosU\Estructuras de datos\L2\P11\cmake-build-debug\P11.exe"  
3  
1  
2  
1
```

The status bar at the bottom indicates the compiler is `.clang-tidy`, the file encoding is `UTF-8`, and the indentation is `4 spaces`. The active project is `P11` and the file is `main.cpp`.