



# I Examen Parcial (2da Parte)

EDyA 361075 Carlos Eduardo Sánchez Torres

9 de marzo de 2021



## 1. Suma de primos

Hacer un programa recursivo que devuelva la suma de los números primos de una lista de N enteros.

### 1.1. Código

```
int main() {  
    Stack<int> stack({1,2,3,4,5,6});  
    cout << sumPrimes(stack); // 10  
    return 0;  
}
```

Figura 1: Principal.

```
int sumPrimes(Stack<int> numbers) {  
    if (numbers.isEmpty()) {  
        return 0;  
    }  
    int acc = numbers.pop();  
    if (isPrime(acc)) {  
        return acc + sumPrimes(numbers);  
    }  
    return sumPrimes(numbers);  
}
```

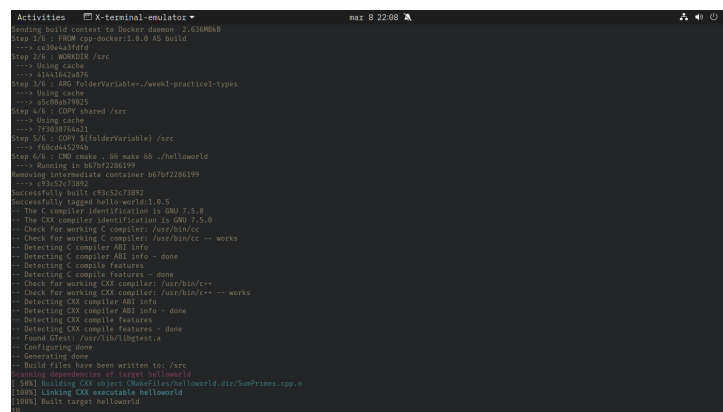
Figura 2: Suma de primos.

```
Stack(initializer_list<int> list) {
    for (auto i = list.begin(); i != list.end(); i++) {
        this->push(*i);
    }
}
```

Figura 3: Crear una pila (lista) desde un arreglo de enteros.

```
bool isPrime(int n) {
    if (n <= 3) {
        return n > 1;
    }
    if (n % 2 == 0 || n % 3 == 0) {
        return false;
    }
    int i = 5;
    while (i*i <= n) {
        int isNDivisibleByPrime = n % i == 0;
        int isNDivisibleByNextPrime = n % (i + 2) == 0;
        if (isNDivisibleByPrime || isNDivisibleByNextPrime) {
            return false;
        }
        i = i + 6;
    }
    return true;
}
```

Figura 4: Determinar si es primo.



```
Activities X-terminal-emulator mar 8 22:08
Downloading build context to Docker daemon 2.610MB/s
Step 1/8 : FROM cyp-docker:1.8.0 AS build
--> 1a3bba1af0e4
Step 2/8 : WORKDIR /src
--> Using cache
--> 41418a2a876
Step 3/8 : ADD felderVariable/.week1/practical/types
--> Using cache
--> 4d58b94911
Step 4/8 : COPY shared /src
--> Using cache
--> f7d0307a4a11
Step 5/8 : COPY ${felderVariable} /src
--> f0d0e5279e
Step 6/8 : CMD cmake . && make && ./helloworld
--> Running in 65bf72281599
Removing intermediate container 6b7bf2286199
--> 09c52c73892
Successfully built 09c52c73892
Successfully tagged helloworld:1.0.5
The C compiler identification is GNU 7.5.0
The CXX compiler identification is GNU 7.5.0
Check for working C compiler: /usr/bin/cc
Check for working C compiler: /usr/bin/cc -- works
Detecting C compiler ABI info
Detecting C compiler ABI info - done
Detecting C compile features
Detecting C compile features - done
Check for working CXX compiler: /usr/bin/c++
Check for working CXX compiler: /usr/bin/c++ -- works
Detecting CXX compiler ABI info
Detecting CXX compiler ABI info - done
Detecting CXX compile features
Detecting CXX compile features - done
Found CMake: /usr/lib/libgtest.a
Configuring done
Generating done
Build files have been written to: /src
Building dependencies of target helloworld
[100] Building CXX object CMakeFiles/helloworld.dir/SunPrimes.cpp.o
[100] Linking CXX executable helloworld
[100] Built target helloworld
```

Figura 5: Resultados.

## 2. Permutaciones

Usando recursividad, hacer un programa que obtenga todas las permutaciones de un conjunto de objetos.

Una permutación de un conjunto de objetos es un orden particular en el que pueden colocarse esos objetos. Cuando hablamos de "todas las permutaciones" de un conjunto, nos referimos a todas las formas posibles de ordenar esos objetos. Ejemplos: Dado el conjunto vacío, la única permutación posible es. Dado el conjunto A, la única permutación posible es A. Dado el conjunto A, B, las posibles permutaciones son AB, BA. Dado el conjunto A, B, C, las posibles permutaciones son ABC, ACB, BAC, BCA, CAB, CBA. Escribir un método que tome un array de caracteres y encuentre todas sus permutaciones.

```

10  int main() {
11      cout << permute(str, "A");
12      cout << "\n";
13      cout << permute(str, "AB");
14      cout << "\n";
15      cout << permute(str, "ABC");
16      return 0;
17  }
18
Terminal: Local x +
[100%] Linking CXX executable helloworld
[100%] Built target helloworld
A
AB
BA
ABC
ACB
BAC
BCA
CAB
CBA
4 files committed: chore(algorithms-test): test2 - permute works (2 minutes ago) 18:1

```

Figura 6: Principal y resultados.

```

string permute(string str, int index = 0, string acc = "") {
    if (index == str.length() - 1) {
        acc += str + "\n";
        return acc;
    }
    for (int subIndex = index; subIndex < str.length(); subIndex++) {
        swap(&str[index], &str[subIndex]);
        acc = permute(str, index + 1, acc);
    }
    return acc;
}

```

Figura 7: Algoritmo para permutar.

### 3. Expresiones balanceadas

Hacer un programa que determine si los delimitadores (,),,,[,] en una expresión aritmética (e.j. [(5+x)-(y+z)]) están equilibrados.

Ejemplo de expresión correcta: ()(())[()] Ejemplo de expresión incorrecta: ([]) Utiliza una pila para implementar la solución. Considera las siguientes pistas:

Si encontramos un símbolo de apertura [, (, debemos apilarla. Si encontramos un símbolo de cierre ], ), entonces consultamos el elemento que hay en la cima de pila. Si son de distinto tipo, podemos afirmar que la expresión no está balanceada. Si son del mismo tipo, debemos desapilar.

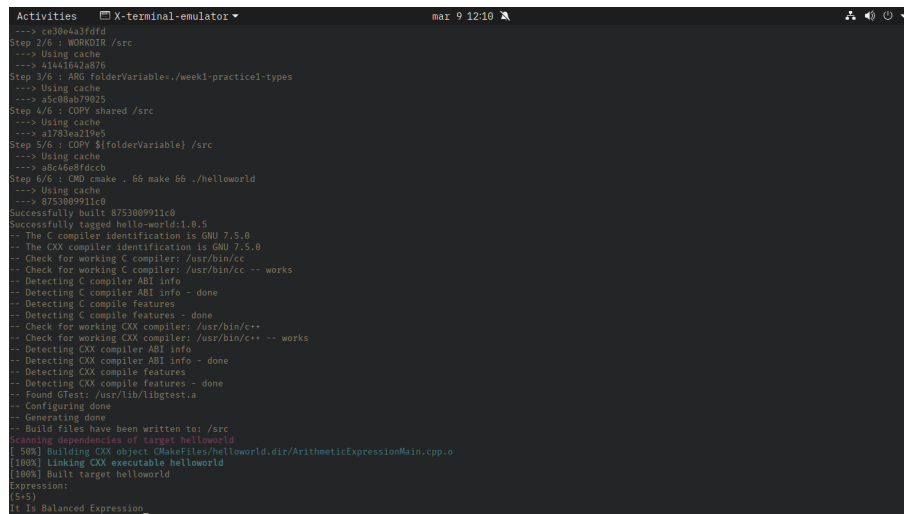
La expresión estará balanceada si al terminar de leer la expresión la pila está vacía.

```
bool isBalancedExpression(string expression) {
    Stack<char> stack;
    for (char &c : expression) {
        if (c == '{' || c == '[' || c == '(') {
            stack.push(c);
            continue;
        }
        if (
            c == '}' && stack.peek() != '{' ||
            c == ']' && stack.peek() != '[' ||
            c == ')' && stack.peek() != '('
        ) {
            return false;
        }
        if (
            c == '}' && stack.peek() == '{' ||
            c == ']' && stack.peek() == '[' ||
            c == ')' && stack.peek() == '('
        ) {
            stack.pop();
        }
    }
    return stack.isEmpty();
}
```

Figura 8: Algoritmo para determinar si la expresión balanceada.

```
int main() {
    string expression;
    cout << "Expression: " << "\n";
    cin >> expression;
    string a = isBalancedExpression(expression) ? "It Is Balanced Expression" : "Is Not Balanced Expression";
    cout << a;
    return 0;
}
```

Figura 9: Principal.



```
Activities X-terminal-emulator mar 9 12:10
--> cd30e4a3fdfd
Step 2/6 : WORKDIR /src
--> Using cache
--> 41441642a876
Step 3/6 : ARG folderVariables ./week1-practice1-types
--> Using cache
--> a5c8ab79825
Step 4/6 : COPY shared /src
--> Using cache
--> a1783e219a5
Step 5/6 : COPY ${folderVariable} /src
--> Using cache
--> a8c4e8dfdc8
Step 6/6 : CMD cmake . && make && ./helloworld
--> Using cache
--> 8753809911c8
Successfully built 8753809911c8
Successfully tagged hello-world:1.0.5
-- The C compiler identification is GNU 7.5.0
-- The CXX compiler identification is GNU 7.5.0
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found GTest: /usr/lib/libgtest.a
-- Configuring done
-- Generating done
-- Build files have been written to: /src
Scanning dependencies of target helloworld
[ 50%] Building CXX object CMakeFiles/helloworld.dir/ArithmeticExpressionMain.cpp.o
[100%] Linking CXX executable helloworld
[100%] Built target helloworld
Expression:
(5+5)
It Is Balanced Expression_
```

Figura 10: Resultados.