

## CS2013 - Programación III

### Simulacro A de Práctica Calificada #2 (PC2)

2023 - 2

Profesor: Rubén Rivas

---

#### Librería Estándar - 7 puntos

Diseñar y desarrollar el template de función **same\_values** que permita identificar si 2 contenedores del mismo tamaño contienen los mismos elementos no importando que sus valores se encuentren en diferente orden. Ejemplo:

```
container_1 = { 1, 4, 5, 19, 20, 11, 22, 1 };  
container_2 = { 1, 1, 4, 5, 11, 19, 20, 22 };
```

#### Casos de uso

```
// Caso de uso #1  
int n = 0;  
cin >> n;  
vector<int> vec1(n);  
vector<int> vec2(n);  
for(auto& item: vec1)  
    cin >> item;  
for(auto& item: vec2)  
    cin >> item;  
cout << same_values(vec1, vec2) << endl;  
// Caso de uso #2  
deque<int> deq1(n);  
list<int> lis1(n);  
for(auto& item: deq1)  
    cin >> item;  
for(auto& item: lis1)  
    cin >> item;  
cout << same_values(deq1, lis1) << endl;
```

### Complejidad Algorítmica - 6 puntos

Dada la siguiente función determinar la complejidad algorítmica, incluir en su respuesta el procedimiento.

```
typedef std::complex<double> Complex;
typedef std::vector<Complex> ComplexVector;

void function (ComplexVector& x) {
    int N = x.size();
    if (N <= 1) {
        return;
    }

    ComplexVector even, odd;
    for (int i = 0; i < N; i += 2) {
        even.push_back(x[i]);
        odd.push_back(x[i + 1]);
    }

    function (even);
    function (odd);

    double theta = 2 * M_PI / N;
    Complex w(1);
    Complex wn(cos(theta), sin(theta));

    for (int k = 0; k < N / 2; ++k) {
        Complex t = w * odd[k];
        Complex u = even[k];
        x[k] = u + t;
        x[k + N / 2] = u - t;
        w *= wn;
    }
}
```

### Programación Concurrente - 7 puntos

Escribir la función template **find\_max\_double\_product** que debe retornar una lista de pares cuyo producto de los 2 valores sea el valor maximo obtenido, la función deberá retornar un vector de pares con todos los valores que cumplan la condición.

#### Caso de uso #1

```
vector<int> vec = generate_container<int, 10000>(-100, 100);  
auto result = find_max_double_product(begin(vec), end(vec));  
cout << result << endl;
```

#### Caso de uso #2

```
list<long int> lst = generate_container<long int, 10000>(-100, 100);  
auto result = find_max_double_product (begin(lst), end(lst));  
cout << result << endl;
```