

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

ESTRUCTURA DE DATOS



BÚSQUEDA BINARIA EJERCICIO 3

NOMBRE: QUISHPE CHAVEZ DENISSE PAULINA

NRC: 29852

FECHA: 02/12/2025

DOCENTE: SOLIS ACOSTA EDGAR FERNANDO

CORTAR CUERDAS EN SEGMENTOS

Tienes N cuerdas de diferentes longitudes (en cm). Necesitas cortarlas para obtener K segmentos de la misma longitud L.

Objetivo: Encontrar la longitud máxima L de los segmentos que te permite obtener al menos K segmentos iguales.

Entrada:

- N: Número de cuerdas
- K: Número de segmentos que necesitas
- Longitudes de cada cuerda (en cm)

Salida: La longitud máxima L (puede ser decimal)

```
#include <iostream>
using namespace std;
double* cuerdas;
int N, K;
// BÚSQUEDA BINARIA: Verificar si con longitud L obtenemos K segmentos
bool puedeCortar(double L) {
    int segmentos = 0;
    for (int i = 0; i < N; i++) {
        // Cuántos segmentos de longitud L sacamos de esta cuerda
        segmentos += (int)(cuerdas[i] / L);
        if (segmentos >= K) {
            return true; // Ya tenemos suficientes
        }
    }
    return segmentos >= K;
}
```

```

}

int main() {
    cout << "==== CORTADOR DE CUERDAS ====\n";
    cout << "Numero de cuerdas: ";
    cin >> N;
    cout << "Segmentos necesarios: ";
    cin >> K;
    cuerdas = new double[N];
    double maxCuerda = 0;
    cout << "Longitudes (cm):\n";
    for (int i = 0; i < N; i++) {
        cout << " Cuerda " << (i + 1) << ":" ;
        cin >> cuerdas[i];
        if (cuerdas[i] > maxCuerda) {
            maxCuerda = cuerdas[i];
        }
    }
    // BÚSQUEDA BINARIA sobre la longitud L (con decimales)
    // Rango: [0.01, cuerda más larga]
    double left = 0.01;
    double right = maxCuerda;
    double respuesta = 0;
    cout << "\n--- BUSQUEDA BINARIA (100 iteraciones) ---\n";
    // Búsqueda binaria con precisión decimal
    for (int iter = 0; iter < 100; iter++) {
        double mid = (left + right) / 2.0;

        if (iter < 10 || iter % 10 == 0) {
            cout << "Iter " << (iter + 1) << ":" L=" << mid << " -> ";
        }
    }
}
  
```

```

}

if (puedeCortar(mid)) {
    if (iter < 10 || iter % 10 == 0) {
        cout << "SI alcanza\n";
    }
    respuesta = mid;
    left = mid; // Intentar con segmentos más largos
} else {
    if (iter < 10 || iter % 10 == 0) {
        cout << "NO alcanza\n";
    }
    right = mid; // Necesitamos segmentos más cortos
}
cout << "\nLongitud maxima: " << respuesta << " cm\n";
// Verificar resultado
cout << "\nVerificacion:\n";
int total = 0;
for (int i = 0; i < N; i++) {
    int segs = (int)(cuerdas[i] / respuesta);
    cout << " Cuerda " << (i + 1) << " (" << cuerdas[i] << "cm) -> "
        << segs << " segmentos\n";
    total += segs;
}
cout << "Total: " << total << " segmentos\n";
delete[] cuerdas;
return 0;
}

```