Imagen que contiene alimentos

Descripción generada automáticamente

Universidad Autónoma del Estado de México

Unidad Académica Profesional Tianguistenco

**Ingeniería en software**

**Unidad de aprendizaje:**

Programación paralela

**Suma de subconjuntos**

**Profesor:**

Gustavo Gómez Vergara

**Alumno:**

Leandro Gómez Flores

Fecha de entrega: viernes, 18 de diciembre de 2020

**Índice**

[Problemática 3](#_Toc58698519)

[Análisis 3](#_Toc58698520)

[**Requerimientos funcionales:** 3](#_Toc58698521)

[**Requerimientos no funcionales:** 3](#_Toc58698522)

[**Tipo de datos:** 4](#_Toc58698523)

[**Caso de uso:** 4](#_Toc58698524)

[**Metodología:** 4](#_Toc58698525)

[**Tipo de caso a los que me podría enfrentar:** 4](#_Toc58698526)

[**Cronograma de actividades:** 5](#_Toc58698527)

[Diseño 6](#_Toc58698528)

[**Tipo de programación:** 6](#_Toc58698529)

[**Lenguaje de programación:** 6](#_Toc58698530)

[**Estructuras de control:** 6](#_Toc58698531)

[**Prototipo de interfaz:** 7](#_Toc58698532)

[**Interfaz:** 8](#_Toc58698533)

[Pruebas 9](#_Toc58698534)

[**Tabla de pruebas con diferentes longitudes de conjuntos:** 9](#_Toc58698535)

[**Imágenes de las pruebas:** 10](#_Toc58698536)

# **Problemática**

Dado un conjunto cuya longitud sea n, sacar todos los subconjuntos posibles cuya suma sea 0, omitiendo así el subconjunto vacío {} y los subconjuntos cuya longitud sea 1: {-1}.

# **Análisis**

## **Requerimientos funcionales:**

* Dar un conjunto aleatorio cuya longitud sea aleatorio dado solo su rango máximo ej. 3 a 15.
* Dar un conjunto aleatorio cuya longitud sea dada por el usuario.
* Adaptable a Programación paralela.
* Se crean ciclos no infinitos.
* Se crean todas las combinaciones posibles de subconjuntos omitiendo así el subconjunto vacío y los subconjuntos cuya longitud sea 1.
* Verificar el tiempo de ejecución cuando se inicie la creación de los subconjuntos y cuando finalice.
* Imprimir en pantalla todos los subconjuntos.
* Imprimir en pantalla todos los subconjuntos cuya suma sea 0.
* Imprimir el numero total de subconjuntos y el numero total de subconjuntos que suman 0.

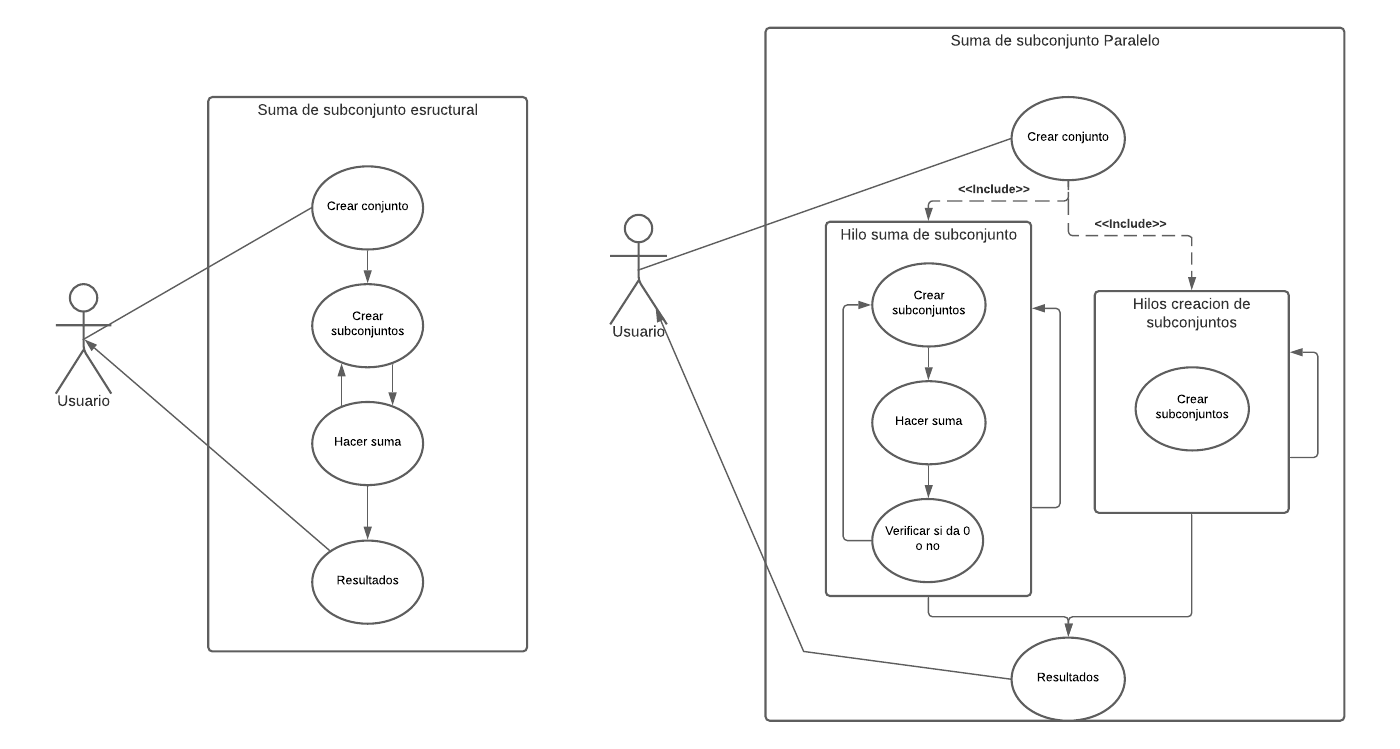
## **Requerimientos no funcionales:**

* Imprimir los subconjuntos de longitud mayor a la longitud menor.
* Tener un tiempo de respuesta aceptable.
* Se mostrará en una interfaz.
* Nombre de variables aceptables.
* Color de interfaz.
* Seleccionar entre el programa echo en paralélelo o Estructural.

## **Tipo de datos:**

|  |  |
| --- | --- |
| * if * if else * for i, i, i++; * long * BigInteger | * int [] * do while * int * String * ArrayList<> |

## **Caso de uso:**



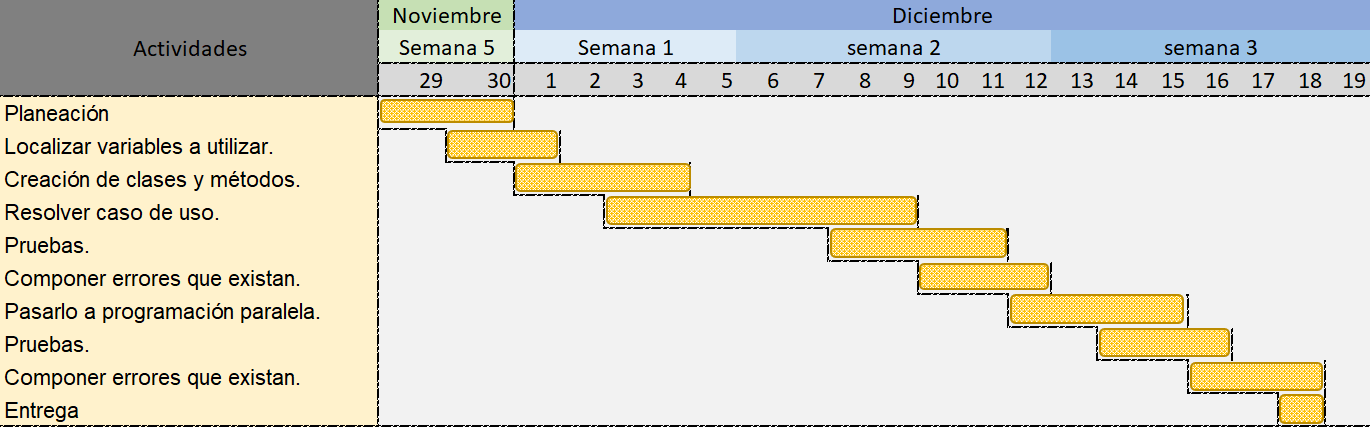
## **Metodología:**

* Incremental.

## **Tipo de caso a los que me podría enfrentar:**

* Tardar mucho en crear subconjuntos.
* Se podrían crear ciclos y no de respuesta.
* Tiempo no óptimo.
* Métodos no optimizados.

## **Cronograma de actividades:**



# **Diseño**

## **Tipo de programación:**

* Programación estructurada.
* Programación paralela.

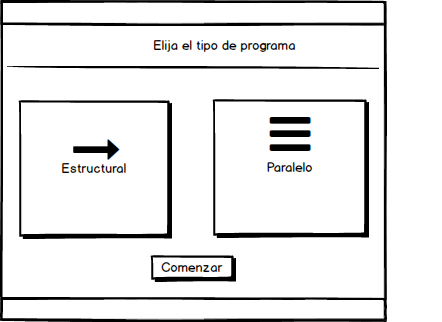
## **Lenguaje de programación:**

* Java

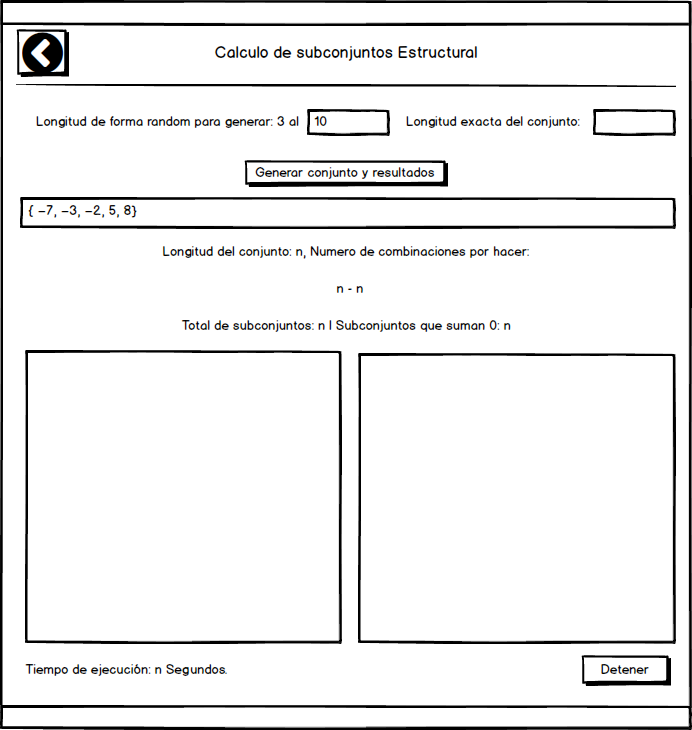
## **Estructuras de control:**

|  |  |
| --- | --- |
| if(condición)  {  Instrucción;  }  if(condición)  {  Instrucción;  }else  {  Instrucción;  } | while(condición)  {  Instrucción;  }  for(int i = N; i>=n; i++)  {  Instrucción;  } |

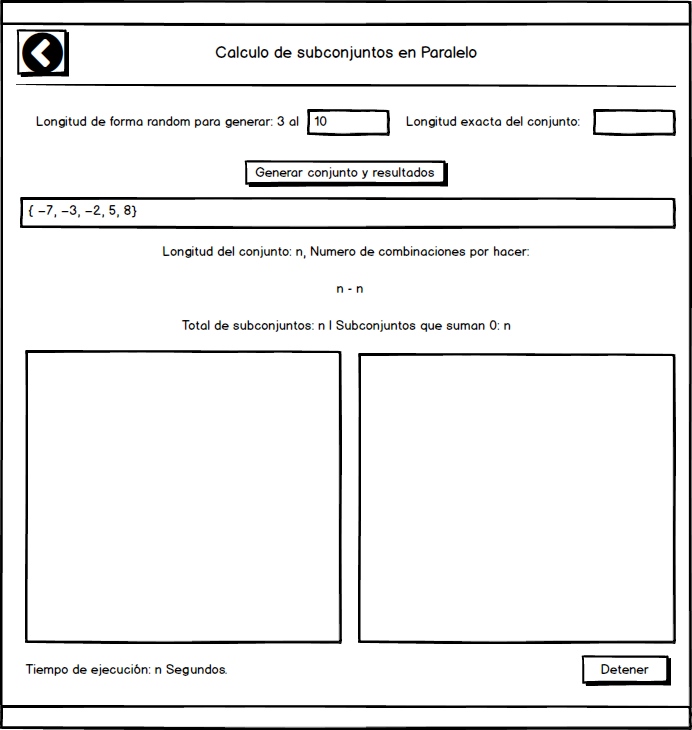
## **Prototipo de interfaz:**



Prototipo 1. Selección de programa



Prototipo 2. Programa Programado de manera estructural

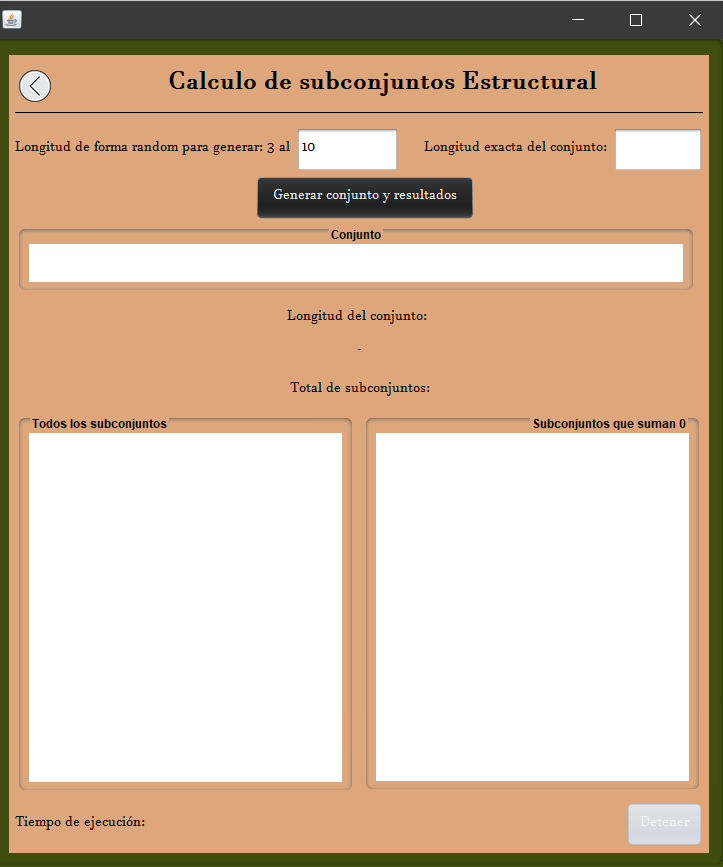


Prototipo 3. Programa realizado paralelo

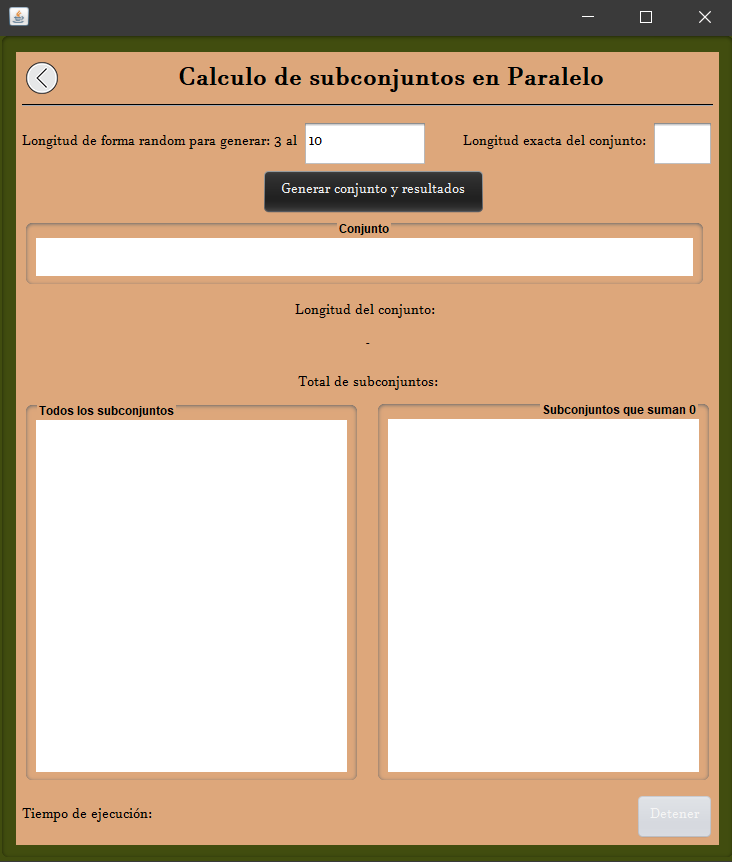
## **Interfaz:**



Interfaz 1. Selección de programa



Interfaz 2. Programa Programado de manera estructural



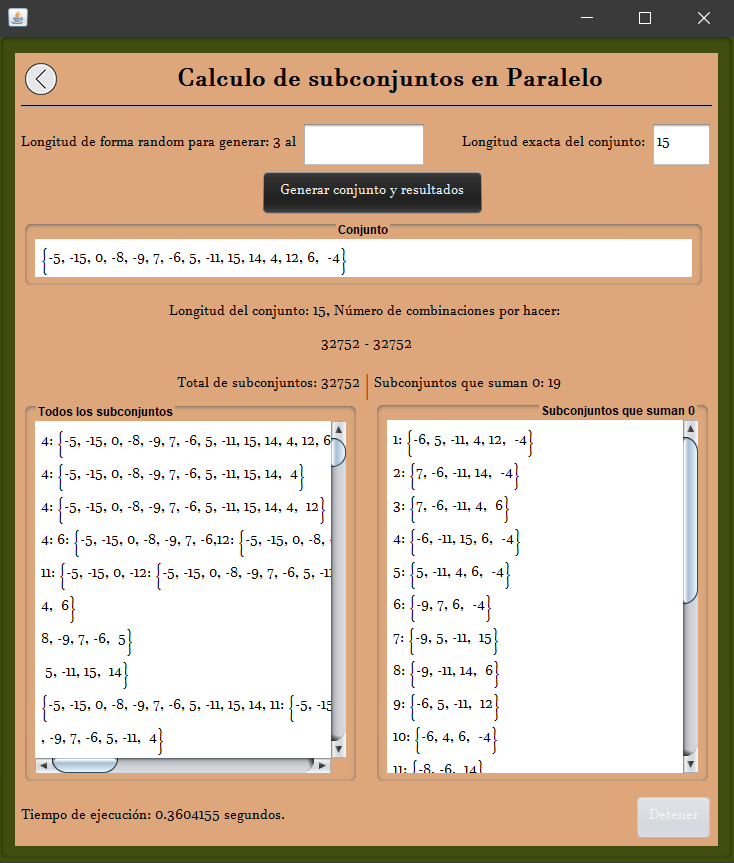
Interfaz 3. Programa realizado paralelo

# **Pruebas**

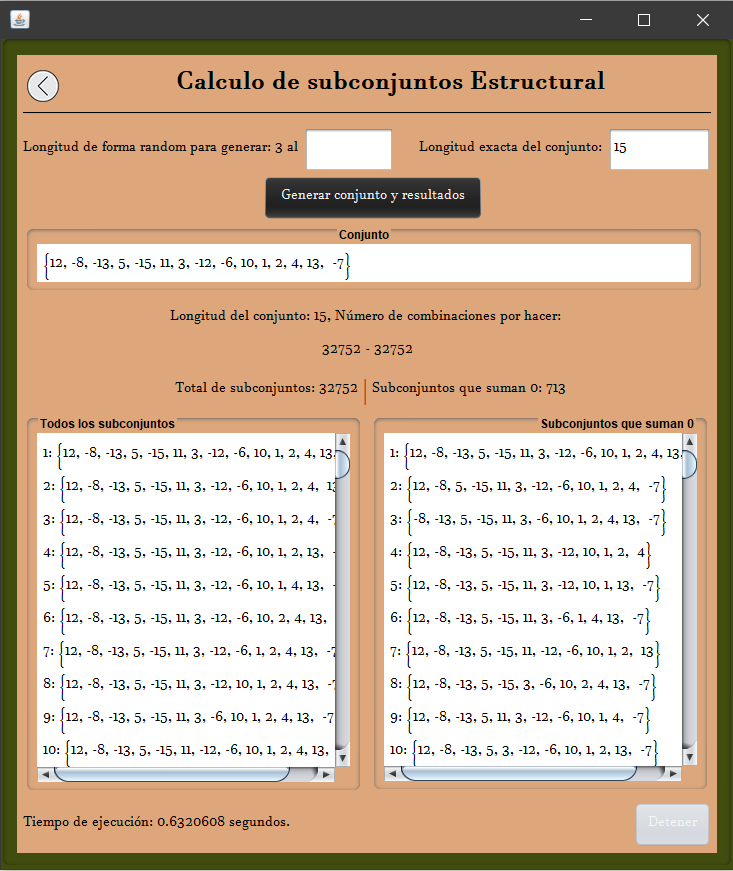
## **Tabla de pruebas con diferentes longitudes de conjuntos:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Programa estructural | | |
| Intentos realizados | Longitud del conjunto | Subconjuntos generados | Tiempo de ejecución |
| 1 | 10 | 1013 | 0.1077434 |
| 2 | 10 | 1013 | 0.0491865 |
| 3 | 10 | 1013 | 0.0332959 |
| 1 | 15 | 32752 | 0.9788268 |
| 2 | 15 | 32752 | 0.7559559 |
| 3 | 15 | 32752 | 0.6320608 |
| 1 | 20 | 1048555 | 37.0297775 |
| 2 | 20 | 1048555 | 34.5920467 |
| 3 | 20 | 1048555 | 30.5967772 |
| 1 | 30 | 1073741793 | 12 min y 0.5821183 seg. |
|  | Programa paralelo | | |
| Intentos realizados | Tamaño del conjunto | Subconjuntos generados | Tiempo de ejecución |
| 1 | 10 | 1013 | 0.226232 |
| 2 | 10 | 1013 | 0.0199908 |
| 3 | 10 | 1013 | 0.0178606 |
| 1 | 15 | 32752 | 0.4975943 |
| 2 | 15 | 32752 | 0.3604155 |
| 3 | 15 | 32752 | 0.4071154 |
| 1 | 20 | 1048555 | 22.8454552 |
| 2 | 20 | 1048555 | 15.0032942 |
| 3 | 20 | 1048555 | 16.6593408 |
| 1 | 30 | 1073741793 | java.lang.OutOfMemoryError: Java heap space. |

## **Imágenes de las pruebas:**



Ejecución del programa paralelo con un conjunto de longitud 15



Ejecución del programa Estructural con un conjunto de longitud 15