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, 11 de diciembre de 2020

**Índice**

[Problemática 3](#_Toc58579165)

[Análisis 3](#_Toc58579166)

[**Requerimientos funcionales:** 3](#_Toc58579167)

[**Requerimientos no funcionales:** 3](#_Toc58579168)

[**Tipo de datos:** 4](#_Toc58579169)

[**Caso de uso:** 4](#_Toc58579170)

[**Metodología:** 4](#_Toc58579171)

[**Tipo de caso a los que me podría enfrentar:** 4](#_Toc58579172)

[**Cronograma de actividades:** 4](#_Toc58579173)

[Diseño 6](#_Toc58579174)

[**Tipo de programación:** 6](#_Toc58579175)

[**Lenguaje de programación:** 6](#_Toc58579176)

[**Estructuras de control:** 6](#_Toc58579177)

[**Prototipo de interfaz:** 7](#_Toc58579178)

# **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.

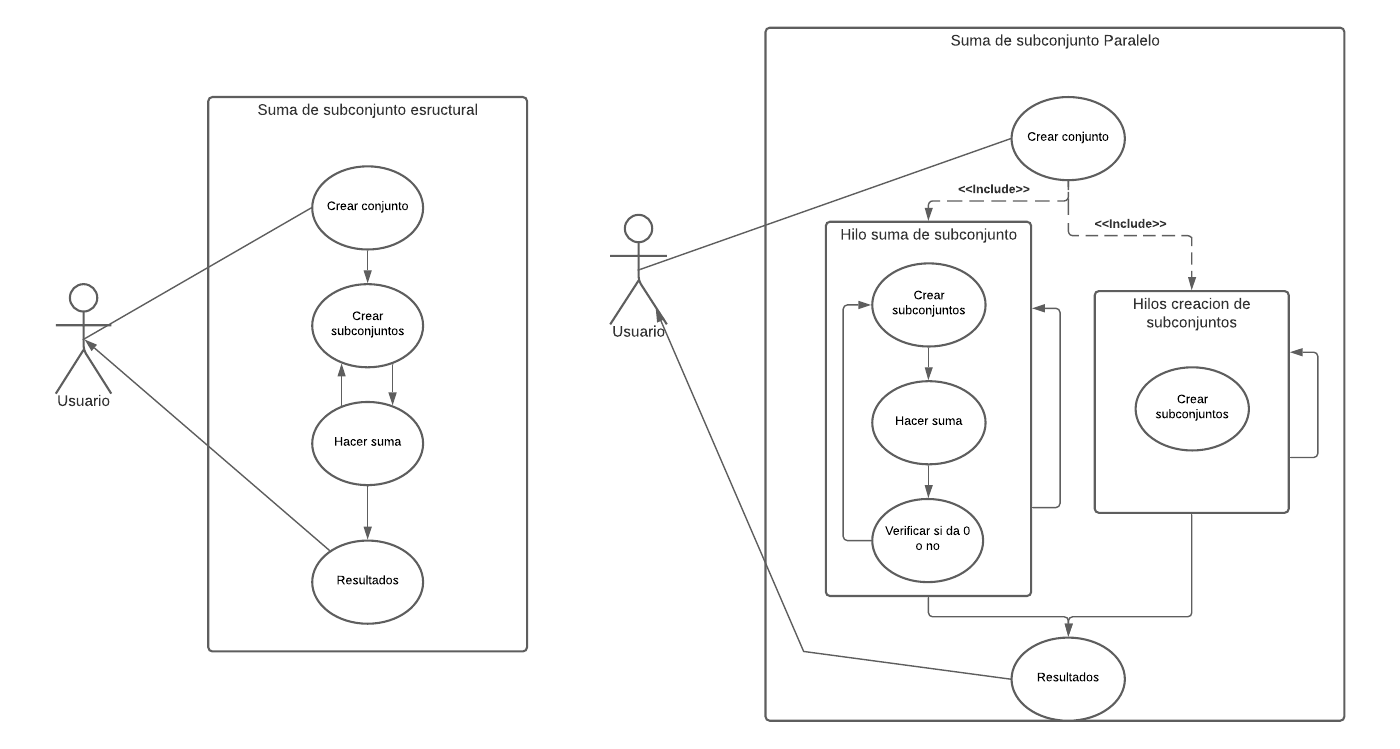
## **Requerimientos no funcionales:**

* 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:**

* Creación de documentación.
* Localizar variables a utilizar.
* Creación de clases y métodos.
* Resolver caso de uso.
* Pruebas.
* Componer errores que existan.
* Pasarlo a programación paralela.

# **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:**

