TAREA INTEGRADORA 3
TEAM JJJ

INTEGRANTES:
JUAN FELIPE CASTILLO GOMEZ
JESUS DAVID RODRIGUEZ BURBANO
JUAN CAMILO RAMIREZ TABARES

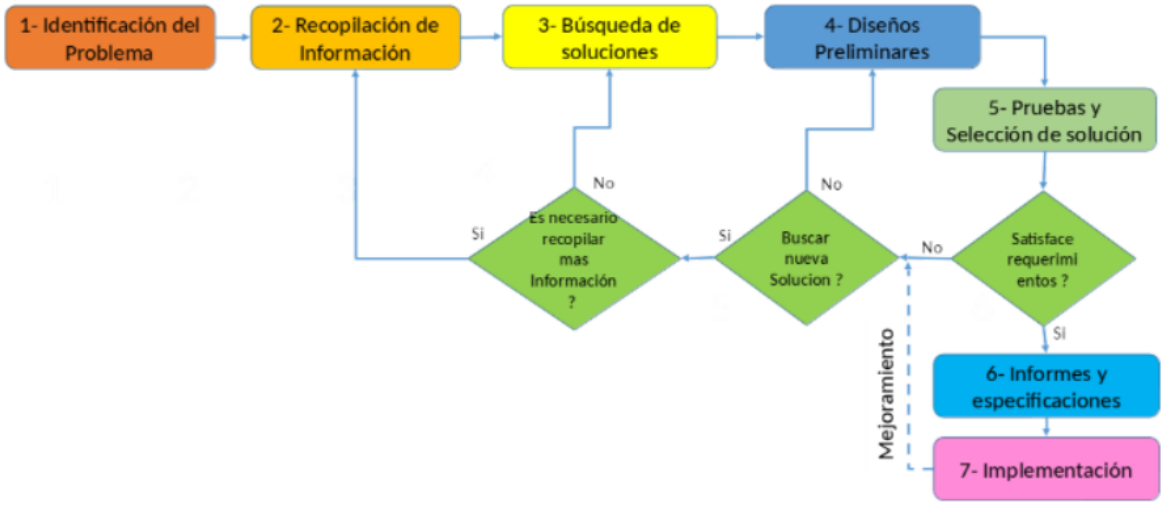
DOCENTE:
JEISON DAVID MEJIA TRUJILLO

COMPUTACIÓN Y ESTRUCTURAS DISCRETAS 1 - GRUPO 1
UNIVERSIDAD ICESI
CALI - VALLE

MÉTODO DE LA INGENIERÍA

Contexto Problemático:
El equipo de profesores de la Facultad de Ingeniería solicita ayuda para la construcción de una red social que permita a las personas de la facultad mantener contacto entre sí. Para ello su solución debe permitir ver las relaciones (contactos) que tiene cada individuo de la facultad, mediante una lista de contactos.

Desarrollo de la Solución:
Para resolver la situación anterior se eligió el Método de la Ingeniería para desarrollar la solución siguiendo un enfoque sistemático y acorde con la situación problemática planteada.
Con base en la descripción del Método de la Ingeniería del libro "Introduction to Engineering" de Paul Wright, se definió el siguiente diagrama de flujo, cuyos pasos seguiremos en el desarrollo de la solución.



Paso 1. Identificación del Problema
Se reconocen de manera concreta las necesidades propias de la situación problemática así como sus síntomas y condiciones bajo las cuales debe ser resuelta.

<u>Identificación de necesidades y síntomas</u>

- Permitir el registro e ingreso de una persona en la aplicación, esta debe contar con su lista de contactos vacía por defecto al momento de ser creada.
    - Cada persona debe contar con un nombre, apellido, correo electrónico, contraseña y departamento de la facultad a la que pertenece.
- Contener un registro de mínimo 50 personas aleatorias previamente registradas en la aplicación.
- Permitir la búsqueda de una persona por su nombre completo.
    - Para este ítem, su solución debe permitir mostrar una lista de coincidencias con la palabra buscada, donde en los primeros resultados se encontrarán las personas que además de coincidir con la palabra buscada tiene al menos un contacto en común con el usuario actual.
- Permitir agregar a una persona a la lista de contactos.

- Mostrar la lista de contactos que tiene cada registro.
- Permitir cambiar de usuario en cualquier momento.
- Desarrollar dos versiones de grafos para el funcionamiento del programa.

Definición del Problema
La facultad de Ingeniería de la Universidad ICESI requiere del desarrollo de un módulo de software que permita observar las relaciones en una red de contactos de la facultad.

Paso 2. Recopilación de Información
Con el objetivo de tener total claridad de los conceptos involucrados se hace una búsqueda de las formas de funcionamiento más utilizadas de redes de contactos.

Definiciones

Grafo
Un grafo en el ámbito de las ciencias de la computación es un tipo abstracto de datos (TAD), que consiste en un conjunto de nodos (también llamados vértices) y un conjunto de arcos (aristas) que establecen relaciones entre los nodos. El concepto de grafo TAD desciende directamente del concepto matemático de grafo.

Red social
En un sentido básico, una red social se basa en establecer relaciones entre personas, que a su vez establecen relaciones entre ellas, creando un grafo de personas unidas por amistad. A través de ellas, se crean relaciones entre individuos o empresas de forma rápida, sin jerarquía o límites físicos.

Paso 3. Búsqueda de Soluciones Creativas
Alternativa 1. Grafo conexo
Un grafo conexo o conectado es un grafo en que todos sus vértices están conectados por un camino(si el grafo es no dirigido) o por un semi-camino (si el grafo es dirigido).

Alternativa 2. Grafo disconexo
Un grafo no conexo o disconexo es un grafo en que al menos alguno de sus vértices no está conectado por un camino hasta otro vértice.

Alternativa 3. Grafo simple
Un grafo es simple si existe una arista uniendo dos vértices cualesquiera. Esto es equivalente a decir que una arista cualquiera es la única que une dos vértices específicos. Un grafo que no es simple se denomina multigrafo.

Alternativa 4. Grafo Dirigido
Un grafo dirigido o digrafo es un tipo de grafo en el cual las aristas tienen un sentido definido, a diferencia del grafo no dirigido, en el cual las aristas son relaciones simétricas y no apuntan en ningún sentido.

Alternativa 5. Multigrafo

Son grafos con dos o más aristas que pueden conectar a un mismo vértice

Alternativa 6. Grafo Bipartito
Son grafos que se pueden dividir en dos subconjuntos disjuntos de vértices,
donde cada una de las aristas conecta un vértice del primer conjunto con uno del segundo.

Paso 4. Transición de las Ideas a los Diseños Preliminares
Para hacer uso de un tipo específico de grafo que modelará de manera correcta la red social, tuvimos en cuenta distintos aspectos tales como: Las relaciones que tienen algunos vértices por defecto, los peores y mejores casos en las relaciones que puede tener nuestro grafo. Y por último, las características que debe tener nuestra red social.

Descarte de ideas:
Conexo
Se descarta el uso de un grafo conexo por el hecho de que no es estrictamente necesario que los nuevos vértices que se creen tengan alguna relación existente con otro vértice.
Multigrafo
Se descarta el uso de un multigrafo debido a que no es posible que un par de vértices tengan más de una arista entre sí.
Bipartito
Se descarta el uso de un grafo bipartito dado que no es posible asegurar la completa definición del mismo en todo momento, debido a los requerimientos del problema.

Paso 5. Evaluación y Selección de la Mejor Solución
Es el momento de seleccionar la opción final, es importante definir criterios de evaluación que nos permitan probar que la solución que seleccionamos es óptima (para nosotros) para este problema en particular pensando en general.

Criterios

Criterio A: Resuelve el problema correctamente.
- [2] Sí
- [1] No

Criterio B: La solución proporciona coherencia (Cuando se hacen amigos las personas).
- [3] Sí
- [2] A veces
- [1] No

Criterio C: La solución tiene en cuenta que las personas que ya son amigos, esto quiere decir que no pueden ser amigos de nuevo.
- [3] Sí
- [2] Mas o menos.
- [1] No

Evaluación

	Criterio A	Criterio B	Criterio C	Total
Alternativa 2. Grafo disconexo	2	3	2	7
Alternativa 3. Grafo simple	2	3	3	8
Alternativa 4. Grafo Dirigido	2	1	2	5

Selección

De acuerdo con la evaluación anterior se debe seleccionar la Alternativa 2 y 3, ya que obtuvieron la mayor puntuación de acuerdo con los criterios definidos.

Paso 6. Preparación de Informes y Especificaciones

Especificación del Problema

Problema: Relacionar las personas en el grafo.
Entradas: Usuarios creados.
Salidas: Usuarios con amistades.

Consideraciones

1. Cuando se crea una relación entre dos personas esta debe ser mutua, es decir, si A es amigo de B, no se puede dar el caso de que B no sea amigo de A.
2. No pueden haber relaciones repetidas, es decir, A no puede ser amigo de B dos veces.
3. No es obligatorio que una persona tenga alguna relación con otro, es decir, A puede o no tener amigos.

Paso 7. Implementación del Diseño

Lista de Tareas a implementar:

- Crear un grafo.
- Calcular la lista de contactos con el algoritmo BFS.
- Buscar relaciones cercanas de un usuario con algoritmo Dijkstra's.
-

Especificación de subrutinas

a:

Nombre:	Grafo
Descripción:	Crear un grafo con vértices y aristas.
Entrada:	
Salida:	

```java
public class Grafo <K>{

  private HashMap<K, Vertice<K>> h;
  private List<Vertice<K>> lv;

  private final boolean dirigido;

  public Grafo(boolean d) {
    dirigido = d;
    h = new HashMap<>();
    lv = new ArrayList<>();
  }

  public boolean newVertice(K e, List<K> ady, List<K> adyDesde) {
    Vertice<K> v = new Vertice<>(e);
    h.put(e, v);
    lv.add(v);
    if(dirigido == true) {
      if(ady != null) {
        for (K k : ady) {
          if(h.get(k) != null) {
            v.insertD(h.get(k));
          }
        }
      }
```

b:

Nombre:	BFS
Descripción:	Encuentra los posibles caminos hacia todos los vértices a partir de uno que se toma como origen.
Entrada:	Un vértice origen.
Salida:	

```java
public void BFS(Vertice<K> v) {
  for (Vertice<K> vertice : lv) {
    vertice.setColor("WHITE");
    vertice.setDistance(Double.MAX_VALUE);
  }

  v.setColor("GRAY");
  v.setDistance(0);
  v.setParent(null);

  Queue<Vertice<K>> q = new LinkedList<>();
  q.add(v);

  while(q.isEmpty() != true) {
    Vertice<K> u = q.poll();
    u.BFS(q);
    u.setColor("BLACK");
  }

}
```

c:

Nombre:	Dijkstra
Descripción:	Calcular el peso mínimo entre el origen de cada usuario
Entrada:	- El vértice origen - El grafo
Salida:	Peso mínimo entre un vértice origen y cada vértice

Referencias

https://es.wikipedia.org/wiki/Grafo_dirigido

https://es.wikipedia.org/wiki/Grafo_conexo

https://www.rdstation.com/es/redes-sociales/#:~:text=Las%20redes%20sociales%20son%20estructuras,sin%20jerarqu%C3%ADa%20o%20l%C3%ADmites%20f%C3%ADsicos.

https://es.wikipedia.org/wiki/Grafo_(tipo_de_dato_abstracto)#:~:text=Un%20grafo%20en%20el%20%C3%A1mbito,del%20concepto%20matem%C3%A1tico%20de%20grafo.
https://posgrados.inaoep.mx/archivos/PosCsComputacionales/Curso_Propedeutico/Matematicas_Discretas/Capitulo_4_Grafos.pdf.