

Novática, revista fundada en 1975 y decana de la prensa informática española, es el órgano oficial de expresión y formación continua de **ATI** (Asociación de Técnicos de Informática), organización que edita también la revista **REICIS** (Revista Española de Innovación, Calidad e Ingeniería del Software). **Novática** co-edita, asimismo **UPGRADE**, revista digital de **CEPIS** (Council of European Professional Informatics Societies), en lengua inglesa, y es miembro fundador de **UPENET** (**UPGRADE** European **NET**work).

<<http://www.ati.es/novatica/>>
 <<http://www.ati.es/reicis/>>
 <<http://www.cepis.org/upgrade/>>

ATI es miembro fundador de **CEPIS** (Council of European Professional Informatics Societies) y es representante de España en **IFIP** (International Federation for Information Processing); tiene un acuerdo de colaboración con **ACM** (Association for Computing Machinery), así como acuerdos de vinculación o colaboración con **AdaSpain**, **AIZ**, **ASTIC**, **RITSI** e **HispaLinux**, junto a la que participa en **ProInnova**.

Consejo Editorial

Ignacio Aguillo Sousa, Guillem Aínsa González, María José Escalona Cuaremas, Rafael Fernández Calvo (presidente del Consejo), Jaime Fernández Martínez, Luis Fernández Sanz, Didac Lopez Viñas, Celestino Martín Alonso, José Onofre Montes Andrés, Francesc Noguera Puig, Ignacio Pérez Martínez, Andrés Pérez Payeras, Viktu Pons i Colomer, Juan Carlos Vigo López

Coordinación Editorial

Llorenç Pagés Casas <pages@ati.es>

Composición y autoedición

Jorge Llácer Gil de Ramales

Traducciones

Grupo de Lengua e Informática de ATI <<http://www.ati.es/gt/lengua-informatica/>>

Administración

Tomas Brunete, María José Fernández, Enric Camarero, Felicidad López

Secciones Técnicas - Coordinadores

Acceso y recuperación de la Información

José María Gómez Hidalgo (Optenet), <mgomez@yaho.com>

Manuel J. María López (Universidad de Huelva), <manuel.mana@diehsia.uhu.es>

Administración Pública electrónica

Francisco López Crespo (MAE), <flc@ati.es>

Arquitecturas

Enrique F. Torres Moreno (Universidad de Zaragoza), <enrique.torres@unizar.es>

Jordi Tubella Morgadas (DAC-UPC), <jordit@ac.upc.es>

Auditoría SITIC

Marina Touriño Troitiño, <marinatourino@marinatourino.com>

Manuel Palao García-Suelto (ATI), <manuel@palao.com>

Derecho y tecnologías

Isabel Hernando Collazos (Fac. Derecho de Donostia, UPV), <isabel.hernando@ehu.es>

Elena Davara Fernández de Marcos (Davara & Davara), <edavara@davara.com>

Enseñanza Universitaria de la Informática

Cristóbal Pareja Flores (DSIP-UJM), <cpajef@dsip.ujm.es>

J. Ángel Velázquez Turbide (DLSI, URJC), <angel.velazquez@urjc.es>

Entorno digital personal

Andrés Marín López (Univ. Carlos III), <amarin@ui3m.es>

Diego Gachet Pérez (Universidad Europea de Madrid), <gachet@uem.es>

Estándares Web

Encarna Quesada Ruiz (Virati), <encarna.quesada@virati.com>

José Carlos del Arco Prieto (TIC Sistemas e Ingeniería), <jcarco@gmail.com>

Gestión del Conocimiento

Juan Baiget Solé (Cap Gemini Ernst & Young), <jbaiget@ati.es>

Informática y Filosofía

José Ángel Olivas Varela (Escuela Superior de Informática, UCLM), <joseangel.olivas@uclm.es>

Karim Gherab Martin (Harvard University), <kgherab@gmail.com>

Informática Gráfica

Miguel Chover Selles (Universitat Jaume I de Castellón), <chover@lsi.uji.es>

Roberto Vivó Hernández (Eurographics, sección española), <rvivo@dsic.upv.es>

Ingeniería del Software

Javier Dolado Cosin (DLSI-UPV), <dolado@si.ehu.es>

Daniel Rodríguez García (Universidad de Alcalá), <daniel.rodriguez@uah.es>

Inteligencia Artificial

Vicente Boti Navarro, Vicente Julián Inglada (DSIC-UPV), <vbotti.vinglada@dsic.upv.es>

Interacción Persona-Computador

Pedro M. Latorre Andrés (Universidad de Zaragoza, AIPO), <platorre@unizar.es>

Francisco L. Gutiérrez Veja (Universidad de Granada, AIPO), <fgutierrez@ugr.es>

Lengua e Informática

M. del Carmen Ugarte García (ATI), <cuarte@ati.es>

Lenguajes Informáticos

Oscar Belmonte Fernández (Univ. Jaime I de Castellón), <bellm@lsi.uji.es>

Inmaculada Coma Tany (Univ. de Valencia), <inmaculada.coma@uv.es>

Lingüística computacional

Xavier Gómez Guinovart (Univ. de Vigo), <xgg@uvigo.es>

Manuel Palomar (Univ. de Alicante), <mpalomar@dsi.ua.es>

Mundo estudiantil y jóvenes profesionales

Federico G. Mon Trotti (RITSI), <gmon.trode@gmail.com>

Mikel Salazar Peña (Asociación Jóvenes Profesionales, Junta de ATI Madrid), <mikelbo_uni@yahoo.es>

Profesión Informática

Rafael Fernández Calvo (ATI), <rfoalvo@ati.es>

Miguel Santes Gurió (ATI), <msantes@ati.es>

Redes y servicios telemáticos

José Luis Marzo Lázaro (Univ. de Girona), <joseluis.marzo@udg.es>

Juan Carlos López López (UCLM), <juancarlo@uclm.es>

Robótica

José Cortés Arenas (Sopra Group), <jccortes@gmail.com>

Juan González Gómez (Universidad Carlos III), <jian@learobotics.com>

Seguridad

Javier Arellito Bertolin (Univ. de Deusto), <jarellito@eside.deusto.es>

Javier López Muñoz (ETS Informática-UMA), <jlm@lcc.uma.es>

Sistemas de Tiempo Real

Alejandro Alonso Muñoz, Juan Antonio de la Puente Alfaro (DIT-UPM), <aalonso.jpunte@dit.upm.es>

Software Libre

Jesús M. González Barahona (Universidad Politécnica de Madrid), <israel.herraz@upm.es>

Israel Herraz Tabernero (UAJ), <isra@herraz.org>

Tecnología de Objetos

Jesús García Molina (DIS-UM), <jmolina@um.es>

Gustavo Rossi (LPIA-UNLP, Argentina), <gustavo@sol.inf.unlp.edu.ar>

Tecnologías para la Educación

Juan Manuel Dodero Beardo (UC3M), <dodero@inf.uc3m.es>

César Pablo Córcoles Briogio (UOC), <ccorcoles@uoc.edu>

Tecnologías y Empresa

Didac López Vilas (Universitat de Girona), <didac.lopez@ati.es>

Francisco Javier Gantús Sánchez (Indra Sistemas), <jfgantus@gmail.com>

Tendencias tecnológicas

Alonso Álvarez García (TID), <aad@tid.es>

Gabriel Marín Fuentes (Interbits), <gabi@atinet.es>

TID y Turismo

Andrés Aguiayo Maldonado, Antonio Guevara Plaza (Univ. de Málaga), <aguiayo.guevara@lcc.uma.es>

Las opiniones expresadas por los autores son responsabilidad exclusiva de los mismos. **Novática** permite la reproducción, sin ánimo de lucro, de todos los artículos, a menos que lo impida la modalidad de © o copyright elegida por el autor, debiéndose en todo caso citar su procedencia y enviar a **Novática** un ejemplar de la publicación.

Coordinación Editorial, Redacción Central y Redacción ATI Madrid

Padilla 66, 3º dcha., 28006 Madrid

Tfn. 91 4029391; fax. 91 3093685 <novatica@ati.es>

Composición, Edición y Redacción ATI Valencia

Av. del Reino de Valencia 23, 46005 Valencia

Tfn./fax 963330392 <secreval@ati.es>

Administración y Redacción ATI Cataluña

Via Llaletana 46, ppal. 1º, 08003 Barcelona

Tfn. 93 41 25 235; fax. 93 41 27 713 <secregen@ati.es>

Redacción ATI Aragón

Lagasca 9, 3-B, 50006 Zaragoza.

Tfn. fax. 97 62 35 181 <seccara@ati.es>

Redacción ATI Andalucía <secreand@ati.es>

Redacción ATI Galicia <secregal@ati.es>

Suscripción y Ventas <<http://www.ati.es/novatica/interes.html>>, ATI Cataluña, ATI Madrid

Publicidad

Padilla 66, 3º dcha., 28006 Madrid

Tfn. 91 4029391; fax. 91 3093685 <novatica@ati.es>

Imprenta: Derra S.A., Juan de Austria 66, 08005 Barcelona.

Depósito legal: B 15.154-1975 - ISSN: 0211-2124, CODEN NOVAEC

Portada: Luces técnicas - Concha Aras Pérez / © ATI

Diseño: Fernando Agresta / © ATI 2003

editorial

La fuerza de trabajo en Informática en resumen

> 02

Construcción sistemática de edificios del conocimiento

> 02

Llorenç Pagés Casas

Noticias de IFIP

Reunión del TC 13 (Human-Computer Interaction)

> 03

Julio Abascal González

Reunión del Consejo de IFIP

> 06

Ramon Puigjaner Trepal

Coordinación editorial

> 07

monografía

Ingeniería de Sistemas de Aprendizaje Electrónico (En colaboración con UPGRADE)

Editores invitados: José Luis Sierra Rodríguez, Antonio Sarasa Cabezuolo, Demetrios G. Sampson, Kinshuk, Ignacio Aedo Cuevas

Presentación. Diseño y construcción de Sistemas de Aprendizaje Electrónico

> 08

Ignacio Aedo Cuevas, Kinshuk, Demetrios G. Sampson, Antonio Sarasa Cabezuolo, José Luis Sierra Rodríguez

Ingeniería de Sistemas de Aprendizaje Electrónico y esfuerzos de estandarización:

Un caso de estudio relativo a los Almacenes de Objetos Didácticos

> 10

Antonio Sarasa Cabezuolo, Daniel Rodríguez Cerezo, José Luis Sierra Rodríguez

Sistemas PDP de contenidos educativos audiovisuales:

Desarrollo de una solución para la Universidad de La Laguna

> 13

Carina S. González González, David Cabrera Primo, Daniel López Barrios, Antonio Barroso Díaz

Patrones de comunicación en sistemas tutores inteligentes basados en componentes

> 17

Géraldine Ruddeck, Dennis Maciuszek, Martina Weicht, Alke Martens

Sistemas Tutores Inteligentes con SCORM

> 23

Gustavo Soares Santos, Álvaro Reis Figueira

Servicios eLearning en Moodle 2.0

> 29

Miguel A. Conde González, Alberto del Pozo de Dios, Francisco J. García Peñalvo

Un caso práctico de integración de agentes y servicios en entornos de eLearning mediante espacios de tuplas

> 34

Francisco Jurado Monroy, Ana I. Molina Díaz, Miguel A. Redondo Duque, Manuel Ortega Cantero

Hacia la invocación de servicios de e-Learning adaptativo: Aplicación a un simulador de navegación y pesca

> 40

Valérie Monfort, Maha Khemaja

Una metodología de diseño para el aprendizaje ubicuo

> 46

Jihen Malek, Mona Laroussi, Alain Derycke, Henda Ben Ghezala

secciones técnicas

Estándares web

Una aproximación basada en modelos para la ejecución automática de modelos de procesos de negocio

> 51

Valeria de Castro, Javier Fabra Caro, Pedro Álvarez Pérez-Arados, Esperanza Marcos Martínez

Redes y servicios telemáticos

Diseño e Implementación de Video Streaming en Redes de Sensores Inalámbricas

> 57

David Rodenas Herráiz, Antonio Javier García Sánchez, Felipe García Sánchez

Robótica

Arquitecturas de control de robots

> 62

José Cortés Arena

Referencias autorizadas

> 65

sociedad de la información

Programar es crear

Mi número de Erdos (solución)

> 72

Julio Javier Castillo, Diego Javier Serrano

La Forja

Creación de un Clúster de Alta Disponibilidad con software libre (solución)

> 74

Miguel Vidal López, José Castro Luis

asuntos interiores

Socios Institucionales

> 77

Monografía del próximo número: "Business Intelligence"

Julio Javier Castillo, Diego
Javier Serrano

Laboratorio de Investigación de Software
MsLabs, Dpto. Ing. en Sistemas de Informa-
ción, Facultad Regional Córdoba - Universi-
dad Tecnológica Nacional (Argentina)

<jotacastillo@gmail.com>,
<diegojserrano@gmail.com>

El programa presentado a continuación ha sido modelado desde el paradigma orientado a objetos, y en base al mismo, se definen las siguientes clases: *Amigo*, *Grafo* y *Main*. La idea básica para resolver el problema será modelar la red social como un grafo dirigido, y luego aplicar sobre él algún algoritmo que nos permita calcular el número de Erdos de cualquiera de los amigos de la red, respecto de otro conjunto de amigos de la red.

Para resolver este problema se ha elegido el algoritmo de Dijkstra o algoritmo del camino más corto, basándonos en la idea de que una relación de amistad es una arista dirigida de un nodo a otro de la red, y etiquetando todas las relaciones de la red con valor de uno. Así, podríamos situarnos en un nodo X, al que llamaremos nodo base-Erdos, y calcular el número X-Erdos de un conjunto de nodos en base a su relación con el nodo X.

Debido a que este proceso debe realizarse repetidas veces y cambiando la base-Erdos, es que se debe ejecutar el algoritmo de Dijkstra tantas veces como bases Erdos distintas nos exija el problema.

La clase *Amigo* contiene un identificador numérico, un nombre y un conjunto de Amigos con los cuales se mantiene una relación de amistad. Esta clase es la abstracción de un nodo de la red social que se está modelando. Vale notar que la relación de amistad es una relación unidireccional, pues A podría declararse amigo de B, pero no necesariamente B tiene que considerar que A es su amigo.

Tres campos privados adicionales (anterior, distanciaTotal, y finalizado) serán empleados por nuestro algoritmo de camino más corto, al momento de recorrer la red.

La clase *Grafo* contiene un conjunto de nodos, es decir, es la abstracción de nuestra red social. Esta clase es la que nos permitirá crear nodos, y establecer relaciones de amistad entre ellos, como así también, contendrá el método clave *dijkstra* que nos permitirá calcular el camino más corto entre un nodo origen y un nodo destino. Nótese, que la longitud de este camino será el número de Erdos del nodo destino respecto del nodo origen, lo cual es justamente el problema que precisamos resolver.

La clase *Main* contiene el método principal

Mi número de Erdos

El enunciado de este problema apareció en el número 208 de *Novática* (noviembre-diciembre 2010, p. 76).

main, mediante el cual solicitaremos el ingreso de datos de la entrada estándar. Se realiza la lectura de la cantidad de casos de prueba a procesar, la cantidad de miembros que habrá en la red social, y las relaciones de amistad de esa red. Toda esta información es esencial para construir la red social.

Como entrada, se ingresará la cantidad de miembros que serán base del cálculo (base-Erdos), y los miembros de la red a los que se les desea calcular su número de Erdos basado en el miembro de origen. Primero, se lee la cantidad de casos de prueba. Luego, por cada caso de prueba se leen la cantidad de miembros de esa red social (variable miembros), y la cantidad de relaciones entre ellos (variable aristas) todos se guardan convenientemente en estructuras de datos. Seguidamente, se leen cada uno de los miembros y su nombre, esto se refleja agregando un nuevo nodo en nuestro grafo *g* (método: *AgregarAmigo()*), por cada miembro. Luego, las relaciones entre esos miembros son agregados como

nuevas aristas del grafo *g* (método: *AgregarAmistad()*).

A continuación, se lee de la entrada la cantidad de miembros que serán base de cálculo del número de Erdos; y luego, se leen los miembros de la red a los cuales se les calculará su número de Erdos, tomando esa base como pivot. A medida que se van leyendo estos miembros, invocamos al método *dijkstra* que nos permite calcular el camino más corto entre el miembro base y el miembro actual. Nótese que la red fue inicializada con pesos unitarios.

Luego, por cada base que cambiamos necesitamos volver a reinicializar el grafo, volviendo a crear los nodos y las conexiones de la red social. Finalmente, se imprime por pantalla la salida en el formato indicado por el problema. La leyenda “*Infinito*” será impresa por pantalla, en el caso de que no haya relaciones entre esos miembros de la red, es decir cuando esos miembros no se conocen.

```
import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.Scanner;

class Amigo {
    public String nombre;
    public HashSet<Integer> amigos;
    public int idMiembro;
    public Amigo anterior;
    public int distanciaTotal = -1;
    public boolean finalizado;

    public Amigo(int idMiembro, String nombre) {
        amigos = new HashSet<Integer>();
        this.idMiembro = idMiembro;
        this.nombre = nombre;
    }

    public void limpiar() {
        anterior = null;
        distanciaTotal = -1;
        finalizado = false;
    }
}

class Grafo {
    private HashMap<Integer, Amigo> miembros;

    public Grafo()
    {
        miembros = new HashMap<Integer, Amigo>();
    }

    public void agregarAmigo(Integer idMiembro, String nombre) {
        miembros.put(idMiembro, new Amigo(idMiembro, nombre));
    }

    public void agregarAmistad(Integer idAmigo1, Integer idAmigo2) {
```

```

    miembros.get(idAmigo1).amigos.add(idAmigo2);
}

public String obtenerNombre(Integer idAmigo) {
    return miembros.get(idAmigo).nombre;
}

public Amigo dijkstra(Integer amigo1, Integer amigo2) {
    Amigo origen = miembros.get(amigo1), destino = miembros.get(amigo2);

    LinkedList<Amigo> cola = new LinkedList<Amigo>();
    cola.add(origen);
    origen.distanciaTotal = 0;
    Amigo x = null;
    while(!cola.isEmpty()){
        x = cola.removeFirst();
        for(Integer am: x.amigos) {
            Amigo y = miembros.get(am);
            if (!y.finalizado) {
                cola.addLast(y);
                int dt = x.distanciaTotal + 1;
                if (y.distanciaTotal == -1 || dt < y.distanciaTotal){
                    y.anterior = x;
                    y.distanciaTotal = x.distanciaTotal + 1;
                }
            }
        }
        x.finalizado = true;
    }
    return destino;
}

public void reiniciar() {
    for(Amigo x:miembros.values()) x.limpiar();
}

public class Main {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int i,j, miembros, aristas;
        String nombre;
        int idMiembro, origen, destino;
        int cantErdos, baseErdos, miembrosErdos;
        int casos = sc.nextInt();

        for(int c = 0; c < casos; c++) {
            Grafo g = new Grafo();
            miembros = sc.nextInt();
            aristas = sc.nextInt();
            for(i=0; i<miembros; i++) {
                idMiembro = sc.nextInt();
                nombre = sc.nextLine().trim();
                g.agregarAmigo(idMiembro,nombre);
            }

            for(i=0; i<aristas; i++) {
                origen = sc.nextInt();
                destino = sc.nextInt();
                g.agregarAmistad(origen ,destino);
            }

            cantErdos = sc.nextInt(); //cant. de num. de Erdos a calcular
            for(i=0; i<cantErdos; i++) {
                baseErdos = sc.nextInt();
                miembrosErdos = sc.nextInt();
                System.out.println(g.obtenerNombre(baseErdos) + ":");
                for(j=0; j<miembrosErdos; j++) {
                    int idMiembroACalcular = sc.nextInt();
                    Amigo x = g.dijkstra(baseErdos,idMiembroACalcular);

                    int val=-1;
                    while (x != null) {
                        val++;
                        x = x.anterior;
                    }
                    System.out.println(g.obtenerNombre(idMiembroACalcular) +
"="+(val==0?"Infinito": (val)));
                }
                g.reiniciar();
            }
        }
    }
}

```