Guía de Ejercicios para el Examen

CC1001 – 05 Computación I

Profesor: Andrés Muñoz
Auxiliares: Oscar Álvarez / Pedro Valencia

Fecha: 27 de Junio de 2007

TABLA DE CONTENIDOS

ARBOLES ABM (PREGUNTA 3, CONTROL 4, 2001)

ENUNCIADO

Un ABM es un árbol binario que en su raíz guarda el menor valor (y los árboles izquierdo y derecho tienen la misma propiedad). Por ejemplo, el siguiente árbol binario es un ABM:

```
              2
         5          4
     8       5          7
```

a) Escribir un método de encabezamiento:

   `boolean esABM(Nodo x)`

   que entregue true si el árbol binario de raíz x corresponde a un ABM (o false si no)

b) Escribir un método de encabezamiento

   `Nodo juntar(Nodo x, Nodo y)`

   que entregue un nuevo ABM uniendo los ABMs de raices x e y.

SOLUCIÓN

Primero veamos la clase nodo, para que nos quede más clara la solución:

```
class Nodo{
       int valor; Nodo izq, der;
       public Nodo(Int x,Nodo y, Nodo z){valor=x, izq=y; der=z;}
}
```

(a) Ahora escribamos el método que “adivina” si es ABM.

```
boolean esABM(Nodo x) {
       // Caso arbol vacio
       if( x == null )
              return true;
       // Caso arbol sin subarboles
       if(x.izq == null && x.der == null)
              return true;
       // Caso arbol izquierdo vacio
       if( x.izq == null )
              return x.valor <= x.der.valor && esABM(x.der);
       // Caso arbol derecho vacio
       if( x.der == null )
              return x.valor <= x.izq.valor  && esABM(x.izq);
       // Caso general
       return x.valor <= x.izq.valor &&
              x.valor <= x.der.valor &&
              esABM(x.izq) && esABM(x.der);
}
```

(b) Ahora vemos como unimos dos árboles y los convertimos en ABM.

```
Nodo juntar(Nodo x, Nodo y) {
```

```
        // Caso ambos arboles vacios
        if( x==null  &&  y==null )
               return null;
        // Caso arbol izquierdo vacio
        if( x==null )
               return new Nodo(y.valor,juntar(y.izq,null),juntar(y.der,null));
        // Caso arbol derecho vacio
        if( y==null )
               return new Nodo(x.valor, juntar(x.izq,null), juntar(x.der,null);
        // Caso general
        if( x.valor <= y.valor )
               return new Nodo(x.valor,juntar(x.izq,x.der),juntar(y,null));
        else
               return new Nodo(y.valor,juntar(x,null),juntar(y.izq,y.der));
}
```

ÁRBOLES ESPEJO (PREGUNTA 2, CONTROL 4, 2002)

ENUNCIADO

Dos árboles binarios son "espejos" si tienen la forma indicada en el siguiente ejemplo:

```
           A                            A
        /    \                       /    \
      B        C                   C        B
    /        /  \                /  \        \
  D        E      F            F      E        D
```

Escribir un método recursivo que reciba las raíces de dos árboles binarios y entregue true si son espejos o false si no lo son.

SOLUCIÓN

Primero declararemos el Nodo para que se comprenda mejor el método recursivo.

```
class Nodo{
       String valor;
       Nodo izq, der;
}
```

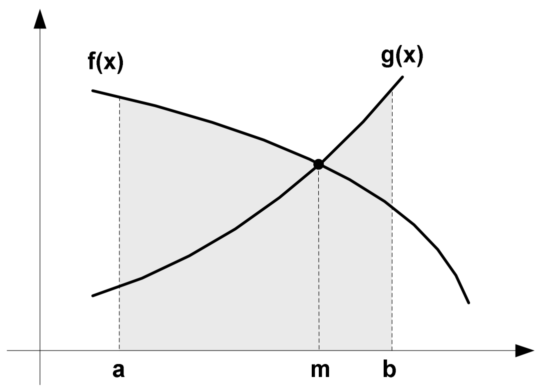
Ahora que ya sabemos la forma del nodo, comezamos a comparar en forma recursiva.

```
boolean espejos(Nodo x, Nodo y) {
       // Caso dos árboles vacíos
       if( x==null && y==null)
              return true;

       // Caso un árbol vacío
       if( x==null || y==null )
              return false;

       // Caso ningún árbol vacío
       return x.valor.equals(y.valor)
              && espejos(x.izq, y.der)
              && espejos(x.der, y.izq);
}
```

INTEGRAL COMBINADA (PREGUNTA 3, CONTROL 4, 2002)

ENUNCIADO

Dadas dos funciones f(x) y g(x) que se intersectan una sóla vez en el intervalo [a,b], escriba un método que entregue una **buena** aproximación del área indicada en el siguiente **ejemplo**:



Notas.

- Invocación del método: area(f, g, a, b, n) en que **n** es el N° total de puntos en que se dividirá el intervalo [a,b]
- Nótese que en el punto **m** se cumple que **f(x)-g(x) = 0**
- Por supuesto, en otros ejemplos, g(x) podría estar por encima de f(x) antes del punto de intersección.

SOLUCIÓN

Veamos como calculamos la integral combinada.

```
static public double area(F f, F g, double a, double b, int n) {
        // Calcular valor de m (raiz de f-g)
        double m = raiz(f,g,a,b,eps);
        // Asegurar que f > g en [a,m]
        if( f.valor(a) < g.valor(a) ) {
                F aux=f;
                f=g;
                g=aux;
        }
        // Calcular area en [a,m]
        double factor = (m-a)/(b-a);
        int n1 = (int)(n*factor);
        double area1 = integral(f,a,m,n1);
        // Calcular area en [m,b]
        int n2 = n -n1;
        double area2 = integral(g,m,b,n2);
        // Devolver area total
        return area1 + area2;
}
```

```java
// Método para calcular raíz de f - g
double raiz(F f, F g, double a, double b, double eps) {
	double x = (a + b)/2;
	if ( b-a <= eps )
		return x;
	if (signo( f.valor(a)-g.valor(a)) == signo(f.valor(x)-g.valor(x)))
		return raiz(f,g,x,b,eps);
	else
		return raiz(f,g,a,x,eps);
}
// Método para saber el signo de la función
int signo(double x) {
	return (x<0):-1:1;
}
// Método para calcular integral
double integral(F f, double a, double b, int n) {
	double s=0, x=a, ancho=(b-a)/n;
	for(int i=1; i<=n; ++i) {
		s += f.valor(x);
		x += ancho;
	}
	return s*ancho;
}
```

Otra solución sería:

```java
double area(F f, F g,double a, double b, int n) {
	double s=0, x=a, ancho=(b-a)/n;
	for(int i=1; i<=n; ++i) {
		s += Math.max(f.valor(x),g.valor(x));
		x += ancho;
	}
	return s*ancho;
}
```

CALENDARIO (PREGUNTA 3, CONTROL 3, 2003)

ENUNCIADO

Los vectores poco densos son vectores en los cuales muchos valores de las componentes v[i] del vector tienen el valor 0. Por esto, se usa guardar en una lista enlazada sólo aquellos valores que son distinto de 0. Cada nodo de esta lista enlazada contiene entonces el índice de un elemento distinto de cero (int) y su valor (double). La clase que implementa un nodo para esta lista es la siguiente:

```
public class Nodo {
    public int indice;
    public double valor;
    public Nodo sig;
    public Nodo(int x, double y, Nodo z) {
             indice = x; valor = y; Nodo = z;   }
}
```

Se le pide programar la clase V con los siguientes métodos:

Nombre	Uso	Explicación
V(double[] x)	V d = new V(valores);	Crea un objeto V con la información que viene en el arreglo pasado como parámetro. En el arreglo, el valor de la mayoría de los elementos es 0
public double pPunto(V x)	Double p = d.pPunto(e);	Retorna el resultado de hacer el producto punto entre los d y e (∑ d[i]*p[i])
public V proyectar(int x)	V e = d.proyectar(indice);	Retorna un objeto V nuevo en donde se han proyectado todos los valores de los elementos del vector salvo el que tiene índice igual al pasado como parámetro

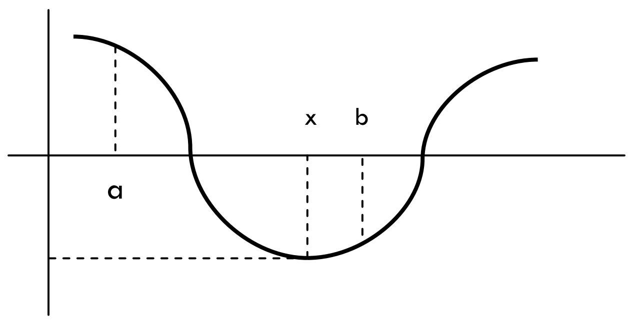
SOLUCIÓN

PENDIENTE

MÍNIMO ABSOLUTO (PREGUNTA 3, CONTROL 4, 2003)

ENUNCIADO

a) Escriba un método que entregue una aproximación (con precisión epsilon) del valor de x que corresponde al único mínimo que toma la función continua f en el intervalo [a,b].



Notas

- Encabezamiento: double xmin(double a, double b, double epsilon, Funcion f). Funcion es una clase o interface que contiene un método de encabezamiento:

  `public double evaluar(double x)`

- Utilice un algoritmo de búsqueda binaria de modo que en cada iteración

  1. encuentre el punto medio (p) del intervalo
  2. evalúe la función en p+epsilon/2 y p-epsilon/2
  3. descarte la primera o segunda mitad del intervalo de acuerdo a si la función aumenta/disminuye en el entorno cercano de p

b) Utilice el método en un programa que calcule el ángulo (con 3 decimales de precisión) en que la función seno toma el menor valor en el intervalo [π,2π].

SOLUCIÓN

(a) Primero veamos como obtener el mínimo absoluto en [a, b]

```
static public double minx(double a,double b,double eps,Funcion f) {
        double p = (a+b)/2;
        if( b-a < eps ) return p;
        double y1 = f.evaluar(p-eps),
               y2 = f.evaluar(p+eps);
        if( y1 > y2 )
               return minx(p,b,eps,f);
        else
               return minx(a,p,eps,f);
}
```

(b) Apliquemos lo anterior para la función Seno. Primero definámosla y luego hagamos el main que calcula el mínimo de la función:

```
interface Funcion {
       public double evaluar(double x);
}

class Seno implements Funcion {
       public double evaluar(double x){
              return Math.sin(x);
       }
}

...
static public void main(String[]args){
       U.println(minx(Math.PI,2*Math.PI,0.001,new Seno()));
}
```

OLIMPIADAS Y MEDALLAS (PREGUNTA 2, CONTROL 3, 2004)

ENUNCIADO

La clase Olimpiada permite manejar los resultados de los juegos a través de los siguientes métodos:

Ejemplo	Resultado	Significado
competencia("tenis masc", "Chile", "Chile", "España")	void	Registrar que el resultado del "tenis masc" fue: oro para "Chile", plata para "Chile", y bronce para "España"
oros("Chile")	int	N° de medallas de oro obtenidas por un país
platas("Chile")	int	N° de medallas de plata obtenidas por un país
bronces("Chile")	int	N° de medallas de bronce obtenidas por un país
pais(10)	String	país que ocupa el lugar 10 en la suma total de medallas
lugar("Chile")	int	Lugar que ocupa Chile en el recuento total de medallas

a) Utilice la clase para mostrar en la pantalla los resultados de las Olimpíadas en la siguiente forma:

```
Lugar  País   Oro    Plata  Bronce Total
1      ...    ...    ...    ...    ...
...    ...    ...    ...    ...    ...
n      Chile  ...    ...    ...    ...
```

b) Escriba los métodos pais y oros considerando que los datos se representan a través de una lista enlazada de nodos con la siguiente información:

```
class Nodo{
       String pais;
       int oros, platas, bronces;
       Nodo sgte;
}
```

Importante. La lista se mantiene ordenada por lugar, es decir, en orden descendente según la cantidad total de medallas obtenidas.

SOLUCIÓN

(a) Solo debemos buscar en el objeto los lugares y las medallas de Chile:

```
Olimpiada O=new Olimpiada();
int l=O.lugar("Chile");
for(int i=1; i<=l; ++i) {
       String pais=O.pais(i);
       int o=O.oros(pais), p=O.platas(pais), b=O.bronces(pais);
       U.println(i+" "+pais+" "+o+" "+p+" "+b+" "+(o+p+b));
}
```

(b) Solo debemos implementar el método según la representación interna entregada:

```
public int oros(String x) {
       Nodo r=primero;
       while(!r.pais.equals(x))
              r=r.sgte;
       return r.oros;
}
```

```
public String pais(int x) {
        Nodo r=primero;
        for(int i=1; i<x; ++i)
                r=r.sgte;
        return r.pais;
}
```

EL TORNEO (PREGUNTA 3, CONTROL 3, 2004)

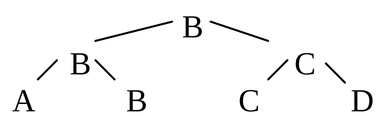
ENUNCIADO

La clase Torneo define las sgtes operaciones para manejar un campeonato por eliminación, es decir, en que el jugador que pierde queda eliminado.

Encabezamiento	Significado
void resultado(String x, String y)	Registrar que el jugador x le ganó al jugador y
int lugar(String x)	Lugar que ocupó el jugador x al final del campeonato. **Nota**. Los valores posibles son 1,2,3,5,9,17,..., 2n +1
String jugador(int x)	Jugador(es) que al final del campeonato ocuparon el lugar x. **Nota**. Si hay más de uno, estarán separados por coma. Si no hay ninguno, se devolverá un String vacío (“”).

a) Escriba un método de encabezamiento void lugares(Torneo T) que muestre la clasificación completa al final del torneo T de 64 jugadores en la forma: 1°X 2°X 3°X,X 5°X,X,X,X 9°X,X,X,X,X,X,X,X 17°... 33°... en que X es el nombre de un jugador.

b) Escriba el método lugar suponiendo que los resultados del torneo se registran en un árbol binario. Por ejemplo, el siguiente árbol binario muestra un torneo entre 4 en que el campeón fue B:




```
class Nodo {
       String valor;
       Nodo izq, der;
}
```

SOLUCIÓN

(a) Lo primero es imprimir la lista completa del torneo en el formato indicado:

```
static public void lugares(Torneo t) {
       U.println("1o"+t.jugador(1));
       for(int i=0; i<=5; ++i) {
              int n=(int)Math.pow(2,i)+1;
              U.println(n+"o"+t.jugador(n));
       }
}
```

(b) Escribamos el método de manera tal que recorramos el árbol buscando en forma recursiva:

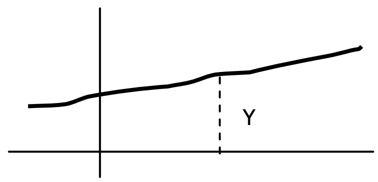
```
public int lugar(String x){
       return lugar(x,1,raiz);
}
protected int lugar(String x,int n,Nodo r){
       if(r==null) return -1;
       if(r.valor.equals(x))
              return n/2+1;
```

```
        return Math.max(lugar(x,2*n,r.izq), lugar(x,2*n,r.der));
    }
```

APROXIMANDO F(X) (PREGUNTA 3, CONTROL 4, 2004)

ENUNCIADO

a) Para una función **f** que es continua y creciente (o decreciente) en el intervalo [a,b], escriba un método que calcule eficientemente el valor de **X** tal que f(X) se aproxime (con precisión epsilon) a un valor **Y** pre-establecido.



Encabezamiento: double X(double Y,double a, double b, double epsilon, Funcion f). Funcion es una clase o interface que contiene el método: double evaluar(double x)

b) Utilice el método en un programa que calcule el valor del coseno del ángulo en que la función seno toma el valor 0.333 ± 0.001 en el intervalo $[0,\pi/2]$.

SOLUCIÓN

(a) Primero que todo, calculemos el X aproximado.

```
double X(double Y,double a,double b,double eps,Funcion f) {
        double x = (a+b)/2;
        double y = f.evaluar(x);
        if( Math.abs(Y-y) <= eps )
                return x;
        if( f.evaluar(a) < y )
                return y<Y? X(Y,x,b,eps,f) : X(Y,a,x,eps,f);
        else
                return y>Y? X(Y,x,b,eps,f) : X(Y,a,x,eps,f);
}
```

(b) Utilizando lo anterior, podemos definir:

```
interface Funcion{
        public double evaluar(double x);
}

class Seno implements Funcion {
        public double evaluar(double x) {
                return Math.sin(x);
        }
}
```

Y por último un programa principal que calcula lo pedido:

```
U.println(Math.cos(X(0.333, 0, Math.PI/2,   0.001, new Seno()));
```

TABLAS DE STRINGS (PROBLEMA 2, CONTROL 3, 2005)

ENUNCIADO

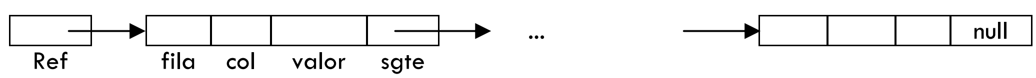
La siguiente clase define los métodos para trabajar con una tabla de Strings, dispuestos en filas y columnas (ambas numeradas de 1 en adelante).

Ejemplo	Significado
Tabla( )	inicializar una tabla que admita cualquier número de filas y columnas
String getValor (int i,int j)	Entregar el valor del elemento ubicado en la fila i y columna j (si no existe entregar el valor null)
void setValor (String x,int i,int j)	Asignar el valor x al elemento ubicado en la fila i y columna j (si no existe, asignarlo por primera vez)

a) Utilice la clase en un método, de encabezamiento **Tabla crearTabla** (), que construya y entregue como resultado (sin mostrar en pantalla) la siguiente tabla:

	1	2	3	...	21
1		1	2	...	20
2	A				
3	B				
...	...				
27	Z				

b) Escriba completamente la clase Tabla considerando que los objetos de la clase se representan a través de una lista enlazada (que sólo contiene nodos para los valores presentes en la tabla):



SOLUCIÓN

(a) Vamos a crear un método que cree la tabla como objeto y lo retorne:

```
static public Tabla crearTabla () {
        Tabla t=new Tabla();
        for(int j=2; j<=21; ++j)
                t.setValor(""+(j-1), 1,j);
        String mayusculas="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        for(int i=2; i<=27; ++i)
                t.setValor(mayusculas.substring(i-2,i-1),i,1);
        return t;
}
```

(b) Escribamos la clase Tabla por completo:

```
class Tabla {
        private Nodo ref;
        public Tabla() {
                ref=null;
        }
        public String getValor(int i, int j) {
                for(Nodo r=ref; r!=null; r=r.sgte)
                        if(r.fila==i && r.col==j)
```

```
                        return r.valor;
                return null;
        }
        public void setValor(String x,int i, int j) {
                for(Nodo r=ref; r!=null; r=r.sgte)
                        if(r.fila==i && r.col==j) {
                                r.valor=x;
                                return;
                        }
                Nodo r=new Nodo();
                r.fila=i;
                r.col=j;
                r.valor=x;
                r.sgte=ref;
                ref=r;
        }
}
```

OPERADORES LÓGICOS (PREGUNTA 3, CONTROL 3, 2005)

ENUNCIADO

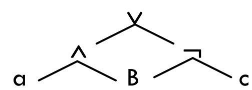
La siguiente tabla muestra el significado de operaciones entre valores lógicos 0 y 1:

a	b	$a \vee b$	$a \wedge b$	$\neg a$	$\neg b$
0	0	0	0	1	1
0	1	1	0	1	0
1	0	1	0	0	1
1	1	1	1	0	0

a) Escriba la clase Logico que implemente las operaciones con los siguientes métodos:

Ejemplos	Resultado	Encabezamiento
Logico a=new Logico(0), b=new Logico(1);	a: referencia a objeto con valor lógico 0 b: referencia a objeto con valor lógico 1	Logico(int x)
a.disyuncion(b)	$a \vee b$	Logico disyuncion(Logico x)
a.conjuncion(b)	$a \wedge b$	Logico conjuncion(Logico x)
a.negacion()	$\neg a$	Logico negacion()

b) Una expresión lógica se puede representar en un árbol binario. Por ejemplo, suponiendo que a, b y c son valores lógicos, la expresión $(a \wedge b) \vee (\neg c)$, se representa por el árbol:



Al respecto, escriba un método de encabezamiento **Logico evaluar(Nodo raiz)** que evalúe el árbol que representa una expresión y entregue el valor lógico del resultado (si está vacío entrega el valor lógico 0).

Nota.

```
class Nodo {
       Object valor;
       Nodo izq, der;
}
```

valor es una referencia a un String (si es un operador) o a un objeto de la clase Logico (si es un operando).

SOLUCIÓN

(a) Escribamos primero la clase Lógico según lo indicado en el eunciado:

```
public class Logico{
       private int v;
       public Logico(int x) {
              v=x;
       }
       public Logico negacion() {
              return new Logico(v==0?1:0);
```

```
        }
        public Logico conjuncion(Logico x) {
               return new Logico(v==1&&x.v==1?1:0);
        }
        public Logico disyuncion(Logico x){
               return new Logico(v==1||x.v==1?1:0);
        }
}
```

(b) Y ahora utilicemos la clase para construir el método evaluar.

```
Logico evaluar(Nodo raiz) {
        // Caso arbol vacio
        if(raiz==null)
               return new Logico(0);
        // Caso operando
        if(raiz.izq==null && raiz.der==null)
               return (Logico)raiz.valor;
        // Evaluar arboles izq y der
        Logico a=evaluar(raiz.izq), b=evaluar(raiz.der);
        // Caso negacion
        if(raiz.valor.equals("~"))
               return b.negacion();
        // Caso conjuncion
        if(raiz.valor.equals("^"))
               return a.conjuncion(b);
        // Caso disyuncion
        return a.disyuncion(b);
}
```

POLINOMIOS DE CHEVYSHEV (PREGUNTA 3, CONTROL 4, 2005)

ENUNCIADO

Los polinomios de Chebyshev se definen por la recurrencia: $T_0(x)=1$; $T_1(x)=x$; $T_{n+1}(x)=2x\,T_n(x) - T_{n-1}(x)$.

Así, $T_2(x)=2x^2-1$, $T_3(x)=4x^3-3x$, y así sucesivamente. Estos polinomios son definidos para cualquier x, pero generalmente son usados en el rango [0,1]. En este intervalo, los polinomios pueden ser calculados con la fórmula $T_n(x)$=coseno(n arcoCoseno(x)).

Considere que arcoCoseno(0)=$\pi/2$ y arcoCoseno(x)=arcoTangente($\sqrt{(1-x^2)}/x$) para x en]0,1].

Por su parte, arcoTangente(x)= $x - x^3/3 + x^5/5 - x^7/7 + \ldots$

SOLUCIÓN

Vamos a definir funciones de forma tal de que podamos ir agrupando cálculos.

```
static public double T(double x,int n) {
       if(n==0) return 1;
       if(n==1) return x;
       if(0<=x && x<=1)
              return Math.cos(n*arcoCoseno(x));
       return 2*x*T(x,n-1)-T(x,n-2);
}

static public double arcoCoseno(double x) {
       if(x==0) return Math.PI/2;
       return arcoTangente(Math.sqrt(1-x*x)/x);
}

static public double arcoTangente(double x) {
       double potencia=x, suma=x, sant=0;
       for(int i=3; Math.abs(suma-sant)>0.00001; i+=2) {
              potencia *= -x*x;
              sant = suma;
              suma += potencia/i;
       }
       return s;
}
```

EL ARBOL GENEALOGICO (PREGUNTA 3, CONTROL 3, 2006)

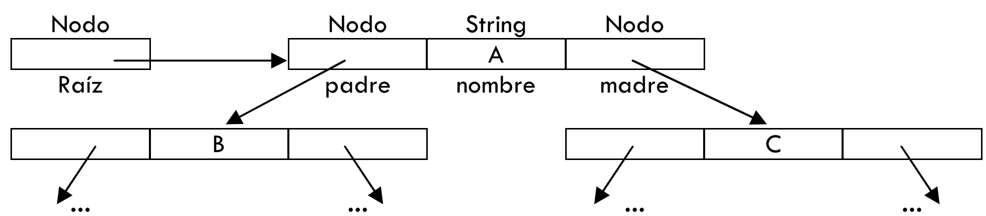
ENUNCIADO

La clase AG permite manejar los datos del árbol genealógico de una persona a través de los métodos:

Ejemplos	Resultado
AG a=new AG("A");	Inicializa árbol genealógico de persona de nombre "A"
a.nombre()	Entrega nombre de la persona
a.agregar("A","B","C")	Registra que el padre de "A" es "B" y que la madre es "C"
a.padre("A")	Devuelve el nombre del padre de "A" (null si no existe)
a.madre("B")	Entrega el nombre de la madre de "B" (null si no existe)

a) Escriba un método de encabezamiento **boolean primos(AG x,AG y)** que utilice la clase AG para entregar true si x e y representan a personas que son primos hermanos (alguno de sus progenitores son hermanos).

b) Escriba el método **madre,** suponiendo que un objeto de la clase AG se representa por el siguiente árbol:



SOLUCIÓN

PENDIENTE

LARGO DE UNA CURVA (PREGUNTA 3, CONTROL 4, 2006)

ENUNCIADO

Escriba un método que entregue una aproximación del largo de la curva de la función f en el intervalo [a,b].

SOLUCIÓN

Para calcular el largo, consideraremos que es equivalente a dividir la recta en puntos que, unidos por rectas, se acercan bastante a la forma de la curva. Definamos entonces el método largo(f, a, b, n) en donde f es un objeto creado a partir de la Interface Funcion, a y b son los extremos del intervalo y n es la cantidad de puntos a utilizar:

```
static public double largo(Funcion f,double a, double b,int n) {
       // Inicializaciones
       double
         ancho=(b-a)/(n-1),
         ancho2=ancho*ancho,
         suma=0,
         x=a,
         yant=f.valor(a);
       // Repetir n-1 veces
       for(int i=1; i<n; ++i){
              // Evaluar la función
              x += ancho;
              double y = f.valor(x);
              // Calcular eficientemente largo de la recta
              double dif = y - yant;
              double l=Math.sqrt(ancho2 + dif*dif);
              yant=y;
              // Actualizar y retornar suma de rectas
              suma += l;
       }
       return suma;
}
```

DE JAVA A MATLAB (BONUS TRACK)

ENUNCIADO

Escriba en Matlab las soluciones para los siguientes problemas ya presentados en esta guía:

a) Integral Combinada (Pregunta 3, Control 4, 2002) [pág 6]

b) Mínimo Absoluto (Pregunta 3, Control 4, 2003) [pág 9]

c) Aproximando f(x) (Pregunta 3, Control 4, 2004) [pág 15]

d) Polinomios de Chevyshev (Pregunta 3, Control 4, 2005) [pág 20]

e) Largo de una Curva (Pregunta 3, Control 4, 2006) [pág 22]

SOLUCIÓN

(a) Primero pongamos la solución en Java:

```
double area(F f, F g,double a, double b, int n) {
        double s=0, x=a, ancho=(b-a)/n;
        for(int i=1; i<=n; ++i) {
                s += Math.max(f.valor(x), g.valor(x));
                x += ancho;
        }
        return s*ancho;
}
```

Luego escribimos la solución en MATLAB:

```
% area(a, b, n): Calcula la integral combinada de F y G (definidas)
function r = area(a, b, n)
x = linspace(a, b, n-1);
fx = F(x);            % Calculamos los y para F
gx = G(x);            % Calculamos los y para G
y = max(fx, gx);      % Calculamos solo los máximos de cada x
suma = sum(y);
r = (b-a) / (n-1) * suma;
```

(b) Primero pongamos la solución en Java:

```
static public double minx(double a,double b,double eps,Funcion f) {
        double p = (a+b)/2;
        if( b-a < eps ) return p;
        double y1 = f.evaluar(p-eps),
                y2 = f.evaluar(p+eps);
        if( y1 > y2 )
                return minx(p,b,eps,f);
        else
                return minx(a,p,eps,f);
}
```

Luego escribimos la solución en MATLAB:

```
% minx(a, b, eps): Calcula el mínimo absoluto de una función F (definida)
function r = minx(a, b, eps)
p = (a + b) / 2;
if b – a < eps
       r = p;
       return;
end
y1 = F(p-eps);
y2 = F(p+eps);
if y1 > y2
       r = minx(p, b, eps);
else
       r = minx(a, p, eps);
end
```

(c) Primero pongamos la solución en Java:

```
double X(double Y,double a,double b,double eps,Funcion f) {
       double x = (a+b)/2;
       double y = f.evaluar(x);
       if( Math.abs(Y-y) <= eps )
              return x;
       if( f.evaluar(a) < y )
              return y<Y? X(Y,x,b,eps,f) : X(Y,a,x,eps,f);
       else
              return y>Y? X(Y,x,b,eps,f) : X(Y,a,x,eps,f);
}
```

Luego escribimos la solución en MATLAB:

```
% X(Y, a, b, eps): Aproxima el valor de F(x) (con F definida)
function r = X(Y, a, b, eps)
x = (a + b) / 2;
y = f(x);
if Y > y
       if Y - y < eps
              r = x;
       elseif f(a) < y & y < Y
              r = X(Y, x, b, eps)
       else
              r = X(Y, a, x, eps)
       end
else
       if y - Y < eps
              r = x;
       elseif f(a) < y & y > Y
              r = X(Y, x, b, eps)
       else
              r = X(Y, a, x, eps)
       end
end
```

(d) Primero pongamos la solución en Java:

```
static public double T(double x,int n) {
       if(n==0) return 1;
       if(n==1) return x;
       if(0<=x && x<=1)
              return Math.cos(n*arcoCoseno(x));
```

```
        return 2*x*T(x,n-1)-T(x,n-2);
}

static public double arcoCoseno(double x) {
        if(x==0) return Math.PI/2;
        return arcoTangente(Math.sqrt(1-x*x)/x);
}

static public double arcoTangente(double x) {
        double potencia=x, suma=x, sant=0;
        for(int i=3; Math.abs(suma-sant)>0.00001; i+=2) {
                potencia *= -x*x;
                sant = suma;
                suma += potencia/i;
        }
        return s;
}
```

Luego escribimos la solución en MATLAB:

PENDIENTE

(e) Primero pongamos la solución en Java:

```
static public double largo(Funcion f, double a, double b, int n) {
        double ancho=(b-a)/(n-1), ancho2=ancho*ancho, suma=0,
                x=a, yant=f.valor(a);
        for(int i=1; i<n; ++i){
                x += ancho;
                double y = f.valor(x);
                double dif = y - yant;
                double l = Math.sqrt(ancho2 + dif*dif);
                yant = y;
                suma += l;
        }
        return suma;
}
```

Luego escribimos la solución en MATLAB:

```
% largo(a, b, n): Aproxima el largo de una curva F (definida) en [a,b]
function r = largo(a, b, n)
ancho = (b-a) / (n-1);
xant = linspace(a, b-ancho, n-2);          % Llenamos con todos menos el ultimo
yant = F(xant);
xpos = linspace(a+ancho, b, n-2);          % Llenamos con todos menos el primero
ypos = F(xpos);
diff = ypos – yant;                        % Calculamos todas las diferencias
larg = sqrt(ancho .^ 2 + diff .^ 2);       % y los largos de cada recta
suma = sum(larg);
r = suma;
```