ICOM4015 Otoño 2001 Exámen I 10/24/01

Nombre: _____

¡Anota tu nombre y número de sección en <u>todas las hojas</u> del examen AHORA!

Tienes 2 horas para completar tres problemas. Lee cuidadosamente todo el examen antes de empezar a trabajar. Muestra todo el trabajo conducente a tu contestación. Podrás recibir crédito parcial por contestaciones parciales siempre y cuando muestres tu trabajo por escrito. Puedes utilizar la parte de atrás de las hojas para cálculos o pseudocódigo. Usa tu tiempo inteligentemente. Suerte.

ICOM 4015 Staff

1	33
2	33
3	33
Total	100

Nombre: _____

Problema 1. (33 puntos) Scope Rules and Parameter Passing Mechanisms

Dado el siguiente código conteste las preguntas **(a)** a la **(d):**

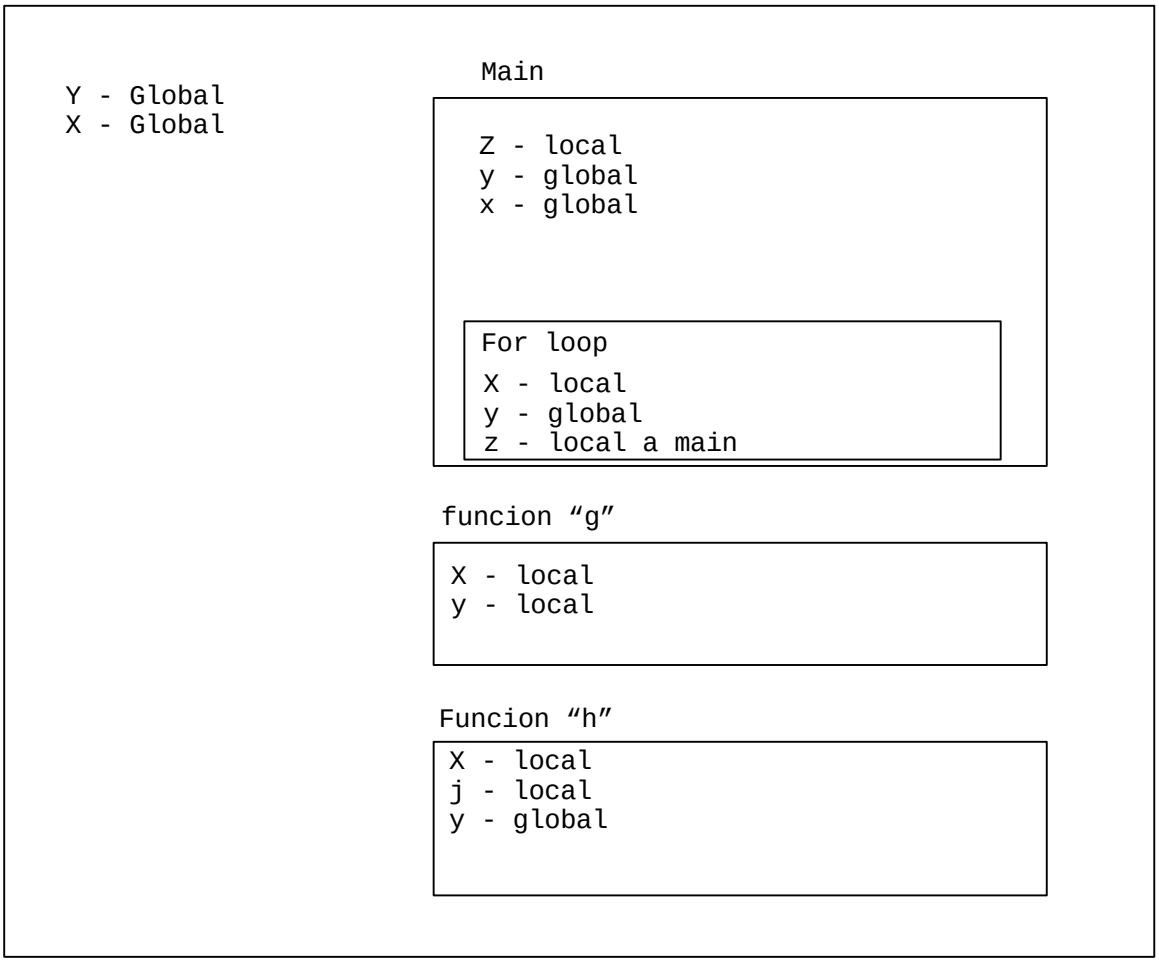
```
#include <iostream.h>

int y = 0;
static int x = 0;

int g(int& x);
int h(int x);

int main() {

      int z = 10;
      y = g(z);
      cout << "y =" << y << " z =" << z << endl;

      x = g(y);
      cout << "x =" << x << " y = " << y << endl;

      for (int x=5; x <= 7; x++) {
            z = h(x);
            cout <<"z =" << z << " x =" << x << endl;
      }

      z = h(x);
      cout << "z =" << z << " y =" << y << " x =" << x << endl;

      return 0;
}

int g(int& x) {
      static int y = 5;
      x += y;
      return y++;
}

int h(int x) {
      static int j = x;
      x++;
      j++;
      y += j;
      return j*x;
}
```

Continúa en la próxima página...

ICOM4015 Otoño 2001 Exámen I 10/24/01

Nombre: _____

(a) (12 puntos) Dibuje el diagrama de bloque, que representa el scope de las variables en el programa descrito, e indique si son globales o locales en el segmento de codigo donde se utilizan.



(b) (13 puntos) Utilizando el diagrama de bloques, traza la ejecución del programa y escribe el output en el espacio provisto:

```
y =5 z =15
x =6 y = 11
z =36 x =5
z =49 x =6
z =64 x =7
z =49 y =39 x =6
```

Continúa en la próxima página...

Nombre: _____

© (4 puntos) Explique cuando se debe usar parámetros por valor:

Se deben usar parámetros por valor cuando no se quiera que cambie el valor del argumento fuera de la función donde se utiliza, excepto cuando el tamaño del objecto pasado como argumento sea considerable.

(d) (4 puntos) Describe concisamente cuando se deben usar parámetros por referencia:

Se debe usar parámetros por referencia cuando se quiera que cambie el valor del argumento fuera de la función donde se utiliza. Esto se hace normalmente para extraer información de la función. También deben utilizarse, cuando el tamaño del objeto pasado como argumento sea considerable.

Nombre: _____

Problema 3. (33 puntos) Procedural Abstraction

El máximo común divisor (MCD) de dos números enteros ***a*** y ***b*** es el máximo elemento del conjunto de los números enteros que dividen tanto a ***a*** como a ***b***. Unos de los algoritmos más antiguos para calcular el MCD se atribuye al filósofo griego Euclides. El algoritmo calcula el residuo de ***a*** y ***b***, llámese ***r***. Si ***r*** es cero, entonces ***b*** es el MCD. De lo contrario el MCD de ***a*** y ***b*** es el MCD de ***b*** y ***r***.

(a) (11 puntos) Escribe una función llamada ***mcd*** que determine el máximo común divisor de dos números enteros utilizando el algoritmo de Euclides. La función debe utilizar ciclos iterativos (e.g. for, while, etc). Especifica su contrato en el espacio provisto para comentarios.

```
// mcd: Devuelve el máximo común divisor de dos números enteros
// Input: Dos números a y b enteros positivos mayores que cero
// Ouput: Maximo comun divisor de a y b
int mcd (int a, int b                    ) {

    int residuo = a % b;
    while (residuo != 0) {
        a = b;
        b = residuo;
        residuo = a % b;
    }
    return b;









}
```

Continúa en la próxima página...

Nombre: _____

(b) (11 puntos) Escribe una función llamada ***sonPrimos*** que utilice la función ***mcd*** para determinar si dos números son relativamente primos. Nos números son relativamente primos si y solo si su máximo común divisor es 1. Especifica su contrato en el espacio provisto para comentarios.

```
// sonPrimos: determina si dos números son relativamente primos
// Input: dos números enteros a y b positivos mayores que cero
// Ouput: cierto (true) si son relativamente primos. Falso (false) de lo
// contratio
bool sonPrimos (int a, int b                          ) {


    return (mcd(a,b) == 1);









}
```

(c) (11 puntos) Escribe una función llamada ***esPrimo*** que utilice la función ***sonPrimos*** de la parte (b) para determinar si un número es primo. Especifica su contrato en el espacio provisto para comentarios.

```
// esPrimo: Determina si un número es primo
// Input: número entero positivo mayor que cero
// Output: Cierto (true) si es primo, falso de lo contrario
bool esPrimo ( int a                             ) {


    int factor = a - 1;
    while((factor > 1) && sonPrimos(a,factor)) {
        factor --;
    }
    return (factor <= 1);
```

```
}
```

Nombre: _____

Problema 3. (33 puntos) Un número perfecto es un entero positivo, que al sumarse todos sus divisores el resultado es igual al mismo numero. Por ejemplo 28 es un número perfecto, ya que la suma de sus divisores es 1+2+4+7+14=28. Si la suma de los divisores es menor que el número, entonces es un número deficiente. Si la suma excede el número entonces es un número abundante. A continuación completarás el siguiente código para que el programa funcione correctamente.

```
//Programa que determina si un número es perfecto,
//Deficiente o Abundante

#include<iostream>

const int perfecto    =1;
const int deficiente  =2;
const int abundante   =3;

void leerDato(int&);
int determinar(int);


int main(){

  int numero;
  leerDato(numero);

  switch(determinar(numero))
    {
    case perfecto :{
      cout<<"Es un numero perfecto.\n";
      break;}
    case deficiente:{
      cout<<"Es un numero deficiente.\n";
      break;}
    case abundante:
      cout<<"Es un numero abundante.\n";
      }
  return 0;
}

void leerDato(int& x){
  do{
    cout<<"Entre un numero entero positivo: ";
    cin>>x;
  }while(x<=0);
}

int divide(int d, int val){
  return val%d==0;
}
```

Continúa en la próxima página ...

ICOM4015 Otoño 2001 Exámen I 10/24/01

Nombre: _____

(a) (5 puntos) Escriba el contrato de la función ***leerDato***:

```
// Lee un número entero (int) positivo de cin
// Si la entrada es un numero negativo debe volver a pedir el numero
```

(b) (5 puntos) Escriba el contrato de la función ***divide***:

```
// Determina si el un numero a es divisible por otro numero b
// Input:  Recibe los dos numeros enteros a y b
// Output: devuelve 1 si el a es divisible por b y 0 de lo contrario
```

Nombre: _____

(c)(15 puntos) Implemente la función ***determinar*** para que cumpla con el contrato descrito.

```
// determinar - Dado un entero positivo x determina si este es perfecto,
// deficiente o abundante. Imprime todos los divisores de x
// así como la suma de estos, además, devuelve un valor
// entero dependiendo del caso: si x es perfecto devuelve 1,
// si x es deficiente devuelve 2 y si x es abundante devuelve 3.

int determinar(int x){


  int suma=0;
  cout << "\nSuma de los divisores: \n ";

  for (int k=1; k<x; k++)
    if(divide(k,x)){
      suma+=k;
      if(k>1) cout<<" + ";
      cout<<k;}
  cout<<" = "<<suma<<endl;
  if (suma==x) return perfecto;
  if (suma<x) return deficiente;
  return abundante;









}
```

Nombre: _____

(d) (8 puntos) Asume que tu programa funciona y cumple a cabalidad con su contrato y que lo has compilado en un archivo ejecutable llamado ***numeros***. Describe cual es el output del siguiente comando.

✍✍**numeros < datos.txt**

Donde **datos.txt** es un archivo con el siguiente contenido:

```
5
6
7
```

datos.txt

Output:

```
[bvelez@coco] numeros < datos.txt
Entre un numero entero positivo:
Suma de los divisores:
 1 = 1
Es un numero deficiente.
```