COMPUTACIÓN 1
Instituto de Computación

Examen - Diciembre de 2006 - 2ª parte

- Duración de esta etapa: 2 Hs.
- No se podrá utilizar ningún tipo de material (apuntes, libro, calculadora, etc). Apague su teléfono celular.
- **Sólo** se contestarán preguntas sobre interpretación de la letra hasta 30 minutos antes de la finalización del mismo.
- Escriba las hojas de un solo lado
- Las partes no legibles del examen se considerarán no escritas
- En la primer hoja a entregar ponga con letra clara, en el ángulo superior derecho, salón en el cual desarrolló la prueba, su nombre, número de cédula de identidad y cantidad de hojas -en ese orden-; las demás hojas es suficiente con nombre, número de cédula y número de página.
- Al entregar su prueba recuerde firmar la planilla correspondiente

Problema 1	30 ptos (15, 15)	

El método de bipartición para encontrar las raíces de una ecuación $f(x) = 0$, donde f es una función continua se basa en la siguiente idea:

Si a y b son dos puntos tales que $f(a) < 0 < f(b)$, entonces f tiene un cero entre a y b.
Para buscarlo, sea c = (a+b)/2.
si f(c) = 0, entonces f tiene un cero en c;
si f(c) > 0, entonces f tiene un cero entre a y c;
si f(c) < 0, entonces f tiene un cero entre c y b.
Repitiendo este proceso se encuentra la raíz de f(x) = 0, con la precisión deseada (i.e. hasta que |a-b| sea menor que un número dado).

Se pide:

a) Implemente una ***función recursiva biparticionR***(a,b,err,f) en ***Matlab*** que resuelva el método de bipartición.
b) Implemente una ***función iterativa biparticionI***(a,b,err,f) en ***Matlab*** que resuelva el método de bipartición.

Observaciónes:

- Utilice el comando de matlab ***feval*** para evaluar la función en el valor c. Por ejemplo para evaluar la función sin de Matlab en el valor 1, se tiene que invocar ***feval('sin',1).*** Para el cabezal de ejemplo se tiene que invocar a ***feval('f',c).***
- Asuma que la función que se le pasa como parámetros posee un cero en el intervalo y cumple que $f(a)<0<f(b)$.
- En caso que el intervalo sea menor que el error, la función debe retornar el extremo menor.

Nota: Se permite utilizar las funciones ***abs*** y ***feval***.
En este ejercicio NO se permite utilizar ninguna función de Matlab que, por su naturaleza, resuelva trivialmente el problema.

Problema 2	25 ptos	

Se desea hallar la representación binaria de un número decimal no negativo N, acotada a un tamaño de R bits (mayor que 0).
Para ello, se pide implementar una ***función iterativa*** *a_binario* en ***Matlab*** que tome como parámetros a N y R y devuelva:

- un número correspondiente al bit de carry, que vale 1 si la representación binaria de N excede la capacidad de bits establecida por R, y 0 en caso contrario

- un vector de largo R con la representación binaria de N. En caso de que ocurra desbordamiento (bit de carry en 1), el vector con la representación binaria debe tomar el resultado tantos bits como lo indique R.

Ejemplos de ejecución:

```
> [c B] = a_binario(12, 4)

c = 0
B = [1 1 0 0]

> [c B] = a_binario(5, 8)

c = 0
B = [0 0 0 0 0 1 0 1]
```

```
> [c B] = a_binario(21, 3)

c = 1
B = [1 0 1]

> [c B] = a_binario(55, 5)

c = 1
B = [1 0 1 1 1]
```

Nota: Se permite utilizar la funcion ***zeros*** de ***Matlab***.
Se pueden utilizar funciones auxiliares pero debe implementarlas usted.
En este ejercicio ***NO*** se puede utilizar la función ***sum***, ni ninguna función ***Matlab*** que, por su naturaleza, resuelva trivialmente el problema.

Problema 3	15 ptos (10,5)	

Se llama *números abundantes* a los números naturales que cumplen ser (estrictamente) menores que la suma de sus divisores sin contarse a sí mismos (divisores propios). Algunos ejemplos de números abundantes son: 12, 18, 24 y 30.

a) Escriba una ***función recursiva*** *sumaDivisores* en ***Matlab*** que reciba como parámetro un entero n y un entero k y devuelva la suma de todos los divisores de n menores o iguales que k. Puede asumir que k es menor o igual que n.

Ejemplos:
sumaDivisores(5, 1) = 1 *ya que el único divisor sería 1*
sumaDivisores(5, 5) = 6 *ya que los divisores serían 1 y 5*
sumaDivisores(5, 0) = 0 *ya que no hay divisores en este caso*
sumaDivisores(10, 6) = 8 *ya que los divisores serían 1, 2 y 5*

b) Escriba una ***función*** *numeroAbundante* en ***Matlab*** que reciba como parámetro un entero y devuelva 1 si es abundante y 0 en caso contrario. Dicha función *debe invocar* a la función escrita en la parte a.

Ejemplos:
numeroAbundante(0) = 0 *ya que no tiene divisores propios*
numeroAbundante(6) = 0 *ya que 1 + 2 + 3 = 6*
numeroAbundante(12) = 1 *ya que 1 + 2 + 3 + 4 + 6 > 12*
numeroAbundante(1) = 0 *ya que no tiene divisores propios*
numeroAbundante(10) = 0 *ya que 1 + 2 + 5 < 10*

Nota: Se permite utilizar la funciones ***rem*** o ***mod y floor*** de ***Matlab***.
En este ejercicio ***NO*** se puede utilizar la función ***sum***, ni ninguna función ***Matlab*** que, por su naturaleza, resuelva trivialmente el problema.