Sección:__ Apellido:_____Nombre:_____Firma:_____
CC1001-Computación I – Control Nº 1: **Pregunta 1** – sábado 18 de agosto de 2007 - Tpo: 1 hr 45 minutos
Con apuntes individuales – Sin consultas - Contestar en estas hojas (si no alcanza, use el reverso)

A)(1.5 punto) Escriba una función que entregue el i-ésimo término de la serie de Leibniz: 1/1, -1/3, 1/5, -1/7, 1/9, …
Por ejemplo, termino(1) entrega 1.0, termino(2) devuelve -0.33…, termino(3) entrega 0.2, etc

```
static public double termino(int i){
  int denom=2*i-1;
  int num=1;
  if(i%2==0)
    num = -1;
  return (double)num/denom;
}
```

B)(1.5 puntos) Use la función anterior en una función **recursiva** que entregue la suma de los primeros n términos de la serie de Leibniz. Por ejemplo, suma(3)=1/1 - 1/3 + 1/5 y suma(4) =1/1 - 1/3 + 1/5 – 1/7.

```
static public double suma(int n){
  //caso base: 0.5
  if(n==0)
    return 0;
  //caso recursivo: 1.0
  else
    return termino(n) + suma(n-1);
}
```

C)(3 puntos) π se puede calcular considerando que $\pi/4 = 1/1 - 1/3 + 1/5 - 1/7 + 1/9$ … Al respecto, escriba un programa que muestre el valor de π sumando 10, 20, 30, … términos (con la función anterior). El programa debe terminar cuando se alcance el valor de π con dos decimales, es decir, cuando los primeros 2 decimales sean 1 y 4.

```
n=10 pi=3.04…
n=20 pi=3.09…
n=30 pi=3.10…
…
n=n° pi=3.14…
```

```
  //repetir para n=10, 20, 30, …: 1.0
  for(int n=10; true; n+=10)
  {
    //calcular y mostrar pi: 1.0
    double pi=4*suma(n);
    U.println("n="+n+" pi="+pi);

    //terminar si 3.14: 1.0
    int i=(int)(pi*100);
    if(i==314) break;
  }
```

Sección:__ Apellido:_____Nombre:_____Firma:_____
CC1001-Computación I – Control Nº 1: **Pregunta 2** – sábado 18 de agosto de 2007 - Tpo: 1 hr 45 minutos
Con apuntes individuales – Sin consultas - Contestar en estas hojas (si no alcanza, use el reverso)

A) (1.5 punto) Escriba una función (recursiva o no) que cuente el nº de veces que un entero X se puede dividir sucesivamente por un entero Y. Por ejemplo: divisiones(12,2) entrega 2, porque 12/2=6 y 6/2=3. Por otra parte, divisiones(12,3)=1, divisiones(12,4)=1,divisiones(12,1) =1, divisiones(12,12) =1, y divisiones(12,0) =0.

```
static public int divisiones(int x,int y){
  if(y<=1) return y;
  int n=0;
  while(x%y == 0){
    x=x/y;
    n=n+1;
  }
  return n;
}
static public int divisiones(int x,int y){
  if(y<=1) return y;
  if(x%y!=0)
    return 0;
  else
    return 1+ divisiones(x/y,y);
}
```

B)(1.5 ptos) Escriba una función que entregue el i-ésimo nº primo. Por ejemplo primo(1) entrega 2, primo(2) devuelve 3, primo(3) entrega 5, …, y primo(nº≤0) devuelve 0. **Nota**. Considere que está disponible la función boolean esPrimo(int x) presentada por el profesor y que devuelve true si x es primo y false si no lo es.

```
static public int primo(int i){
  if(i<=0) return 0;
  if(i==1) return 2;
  int n=1, j=1;
  while(n!=i){
    j = j + 2;
    if( esPrimo(j) )
      n=n+1;
  }
  return j;
}
```

C)(3.0 puntos) Escriba un programa que use las funciones anteriores en un programa que lea una lista de números enteros que termina con un 0 y escriba su descomposición en factores primos.

```
 //repetición hasta un cero: 0.5
  while(true){
    int n=U.readInt("n°?");
    if(n==0) break;
    //generar los factores primos de n: 1.0
    for(int i=1; true; ++i){
       int p=primo(i);
       if(p>n) break;
       //repetir el factor primo p: 1.5
       int d=divisiones(n,p);
       for(int j=1; j<=d; ++j)
         U.print(p+" ");
    }
    U.println("");
  }
```