UNIVERSIDAD DE VALLADOLID **Departamento de Informática**
Informática Licenciatura en Matemáticas
Parcial Febrero de 2009

Apellidos : Nombre :

AVISOS:

- Es necesario entregar todas las hojas del enunciado del examen al finalizar, se hayan respondido o no a las preguntas.
- Es obligatorio firmar todas las hojas de respuesta del examen.
- En breve estará en la página web de la asignatura una copia del enunciado del examen y la solución del mismo.
- **La solución de cada problema debe entregarse en una hoja separada**.
- **La duración de esta parte del examen es de 1 hora y 30 minutos.**

Problemas

1. (**3 puntos**) **Sólo en código algorítmico**

   Crea un programa principal que solicite al usuario un número entero positivo válido, N, y que a continuación le muestre al usuario aquellos números menores o iguales que N que verifiquen que la suma de sus divisores es un número primo.

   El programa debe realizar llamadas a algoritmos con nombre para realizar aquellas operaciones que consideres relevantes. En ese caso, debes aportar las especificaciones y el código de cada uno de los algoritmos con nombre que utilices.

   El programa debe repetir esta operación hasta que el usuario decida terminar.

   Debes indicar de forma razonada el Principio Básico de Resolución y ¿qué tipo/s de esquema/s repetitivo/s estás empleando y por qué lo/s has elegido?

```
PBR: El programa debe repetir varias veces la misma operación: pedir un número
al usuario y analizar los datos entre 2 y ese número. El tratamiento se ealizará
al menos una vez, pero un número indeterminado de veces. Debe usarse el esquema repetir.

El tratamiento que hay que realizar es el mismo para todos los elementos entre 2
y N. Por lo tanto,  debe ser un esquema para.

Ese tratamiento consistirá en calcular los divisores del número, acumulando sus
valores, y a continuación comprobar si el número es primo. Ambas operaciones son
no elementales, pero ya vistas en clase. Se usarán algoritmos con nombre:

- suma_divisores (k, suma): recorre los posibles divisores de k, entre 1 y k/2,
comprobando si lo son. Si es así, los acumula en una variable suma. Conozco
número de iteraciones, esquema para.
- es_primo (k, resp): comprueba si un número es primo, revisando si tiene algún
divisor entre 2 y k/2. Si no lo tiene, es primo. No se conoce el número de
```

iteraciones a priori, esquema mientras.

```
Algoritmo es_primo (k, resp) es
k: numérico; {p. dato}
resp: lógico; {p. resultado}
i: numérico;
Inicio
  i:=2;
  mientras i <= k/2 and k mod i <> 0 hacer
   i := i + 1;
  fin mientras;
  si i > k/2 entonces resp := cierto;
  sino resp := falso;
  fin si;
Fin

Algoritmo suma_divisores (k, suma) es
k: numérico; {p. dato}
suma: numérico; {p. resultado}
i: numérico;
Inicio
  suma := 1;
  para i desde 2 hasta k/2 hacer
   si k mod i = 0 entonces
     suma := suma + i;
  fin para;
Fin

Algoritmo principal es
n: numérico; {dato a leer}
i: numérico; {contador}
suma: numérico; {para calcular la suma de los divisores de N}
resp: carácter; {quiere seguir el usuario o no}
es: lógico; {si el numero cumple las condiciones o no}
Inicio
resp := 's';
repetir
  escribir "Dame un numero para comprobar cuantos numeros especiales hay entre 2 y él:";
  leer n;
  para i desde 2 hasta n hacer
    suma_divisores (i, suma);
    es_primo (suma, es);
    si es entonces
      escribir "los divisores de ", i, " suman ", suma, " que es un numero primo";
  fin para;
  escribir "Quieres continuar (s/n)?";
  leer resp;
hasta que resp = 'n' or resp = 'N';
Fin
```

2. **(1,5 puntos) Función en C.**

Disponemos de un vector de números reales, *notas*, con un máximo de 12 elementos, que almacena los resultados de los exámenes de un/a alumno/a.

Queremos hacer una función que calcule cuáles han sido las notas: más alta, más baja y promedio de ese estudiante, y que nos indique mediante un mensaje por pantalla las posiciones de las notas más alta y más baja dentro del vector.

Debes indicar de forma razonada el Principio Básico de Resolución y ¿qué tipo/s de esquema/s repetitivo/s estás empleando y por qué lo/s has elegido?

La función debe responder a las siguientes especificaciones en código algorítmico:

```
Algoritmo varias_operaciones (notas, N, mas_alta, mas_baja, media, ok) es
notas:vector[1..12] de numérico; {parámetro dato; notas del estudiante;
  estarán almacenadas en las N primeras posiciones}
N: numérico; {entero; parámetro dato; número real de notas almacenadas;
 Estará comprendido entre 1 y 12; en otro caso, será fuente de error}
mas_baja, mas_alta: numérico; {parámetros resultado;
  valores numéricos de las notas más baja y más alta, respectivamente}
media: numérico; {parámetro resultado; valor de la nota media del alumno/a}
ok: lógico; {parámetro resultado;
  indica cierto si se pudieron hacer las operaciones; falso en caso contrario}
```

Debes indicar de forma razonada:

- En primer lugar ¿cómo se transforma esta cabecera en una función en C: qué tipo de función es y cómo se especifican los parámetros en C?,
- en segundo lugar, ¿qué tipo de esquema repetitivo estás empleando y por qué lo haces?

Se valorará la eficiencia de la solución en términos del número de esquemas repetitivos que utilices y del tipo de variables locales que definas.

```
- Para calcular el máximo, mínimo y media de los valores es necesario recorrer
todos los elementos del vector y sobre ellos realizar el mismo tratamiento.
El número de iteraciones es conocido, N, por lo tanto, utilizaremos el esquema para.

Para cada elemento del vector, excepto para el primero, hay que comprobar si es
mayor que el máximo o menor que el mínimo. En todos los casos, hay que acumular
su valor para después calcular la media. Si hay modificación del máximo o del
mínimo, hay que almacenar la posición en la que se encuentran, para mostarlo al
final del algoritmo.

En el algoritmo hay cuatro parámetros resultado. Se elige uno como tipo de la
función (en este caso ok, lógico, se transformará en un entero, int, en C).
El resto se pasarán por referencia, esto es, como punteros.
```

```
int varias_operaciones (notas, N, mas_alta, mas_baja, media)
float notas[]; /* me sirve la función para cualquier vector de reales*/
int N; /* numero real de datos*/
float *mas_alta, *mas_baja, *media; /* parametros resultado */
{
int ok; /* variable local, guardará el valor que devuelva la función*/
int i; /* variable local, contador de las posiciones de notas */
int posmax, posmin; /* posiciones que ocupan el máximo y el minimo */

if (N>0) {
ok = 1;
/* Inicializamos los valores con la primera posicion del vector*/
posmax=0;
posmin=0;
*mas_alta = notas[0];
*mas_baja = notas[0];
*media = notas[0];
for (i=1; i < N; i++) {
  *media += notas[i];
  if (notas[i] > *mas_alta) {
    posmax = i;
    *mas_alta = notas[i];
  }
  else if (notas[i] < *mas_baja) {
    posmin = i;
    *mas_baja = notas[i];
  };
} /* for */
*media /= N;
 printf("\nEl valor maximo %f esta en la posicion %d", *mas_alta, posmax + 1);
 printf("\nEl valor minimo %f esta en la posicion %d", *mas_baja, posmin + 1);
} /* if*/
else
 ok = 0;
return ok;
}
```