Fundamentos de Informática 1ºIIND

Examen Teoría Septiembre 2005

Nombre	**Apellidos:**
Grupo: □ **A** □ **B** □ **C** □ **D** □ **E**	

Preguntas Cortas	**Programar código**			**Nota final**
Preguntas cortas (5 puntos)	Prob.1 (2 punto)	Prob.2 (1,5 puntos)	Prob.3 (1,5 puntos)	

Advertencias:

- Duración del examen 2 horas y media .
- No se puede desgrapar el cuadernillo del examen.
- Se puede utilizar lápiz.
- Las preguntas cortas mal contestadas no descuentan.
- Las funciones sólo pueden tener un `return`
- No se puede utilizar `break` ni `continue`
- No se aconseja utilizar `exit`, sólo se admite en casos de errores excepcionales
- No se pueden utilizar variables globales

Primera Parte: Preguntas Cortas

Página en blanco

Preguntas Cortas (5 puntos)

1.1 Prototipos

Indicar cuáles de los siguientes prototipos son válidos para calcular el número de mayúsculas y el número de minúsculas de una frase.

□ sí	□ no	`char Contar(char *frase, int may, int min);`
□ sí	□ no	`void Contar(char frase[], int *pmay, int *pmin);`
□ sí	□ no	`int Contar(char frase, int *pmay);`
□ sí	□ no	`void Contar(char frase, int longitud, int *may, int *min);`

1.2 Bucles

Escribir el código C de una **función** que reciba una matriz de enteros y que calcule cuántos elementos son pares, cuántos son impares y cuántos son inválidos.

- Son pares los enteros positivos divisibles por 2, es decir cuyo resto al dividir por 2 sea cero.
- Son impares los enteros positivos no divisibles por 2.
- Se consideran inválidos los negativos y el cero.

1.3 Operaciones con enteros

Escribir una función que calcule la letra del NIF. La función debe recibir el DNI en un `long` y devolver un `char` con la letra. La letra del NIF se obtiene calculando el resto de la división por 23 y se elige la letra del siguiente vector tipo `char`:

```
char letra_NIF[]="TRWAGMYFPDXBNJZSQVHLCKE";
```

Por ejemplo, si el resto de la división del DNI por 23 es 3, la letra del NIF es A.

1.4 Funciones

Escribir el prototipo y la llamada a la función `Datos` que inicializa los datos de una persona:

```
typedef struct{
       char nombre[30];
       int telefono ;
}t_persona ;
....................................../*prototipo funcion Datos()*/
int main(void)
{
       t_persona p;

       ................................/*llamada a funcion Datos()*/
       ...
}

....................................../*cabecera funcion Datos()*/
{
       strcpy(p->nombre,"Alberto");
       p->telefono =91101010;
}
```

1.5 Vectores y Matrices

Un programa utiliza una función llamada `LeerDatos()` para leer de teclado los datos de **una** película (Titulo, Tiempo en minutos y Código de referencia). Completa el prototipo y el código de la función `LeerDatos()`:

```
#define N 100
typedef struct {
   char Titulo[80];
   float Tiempo;
   int Codigo;
}t_pelicula;

................................................................/*prototipo*/

int main(void)
{
   t_pelicula  movie[N];

   ...
   LeerDatos(movie,25);  /* lee la película 25 */
   ...
}
```

```
.................................................../*código de LeerDatos()*/
{
```

```
}
```

1.6 Archivos

Escribe una función de tipo `t_persona Fichero(char nom[])` que reciba el nombre de un cliente y abra el archivo binario `clientes.dat`, para buscar los datos de una persona identificada por dicho nombre. La función devolverá la estructura correspondiente a la persona. Cada estructura contiene el campo `nombre` del cliente y otros campos con el resto de los datos.

ADVERTENCIAS:

- La función no puede utilizar `exit` y sólo puede tener un `return`.
- La función no debe utilizar `printf` salvo para mostrar mensajes de error

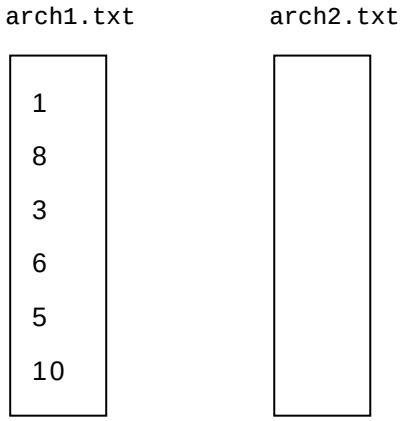
1.7 Archivos

Indica los datos que resultan en `arch2.txt` suponiendo que el archivo `arch1.txt` contiene los valores indicados:

```
void main( )
{
       FILE *archivo1;
       FILE *archivo2;
       int dato ;
       int n;

       archivo1 = fopen("arch1.txt", "r");
       archivo2 = fopen("arch2.txt", "w");

       ...
       n=fscanf(archivo1, “%d", &dato);
       while (n==1) {
          if(dato%2==0) {
             dato = dato * 3;
          }
          fprintf(archivo2, “%d\n", dato);
          n=fscanf(archivo1, “%d", &dato);
       }
       fclose(archivo1) ;
       fclose(archivo2) ;
}
```



1.8 Punteros y Funciones

¿ Qué resultado mostrará el siguiente programa por pantalla:?

```
void incrementa(int *x);
void decrementa(int x);

int main(void )
{
   int x = 12 ;
   incrementa(&x);
   decrementa(x);
   printf(“%d”, x);
}

void incrementa(int *x)
{
   (*x)++ ;
   return ;
}

void decrementa(int x)
{
   --x ;
   return ;
}
```

La salida por pantalla del código es: .....

1.9 Funciones

Escribir el **prototipo** y la **llamada** a la función `Funcion1()` que recibe como argumento un vector de discos y el nombre de un cantante y obtiene el primer disco encontrado de ese cantante y el número de discos totales de ese cantante.

```
/* Prototipo */
.....................................................................

int main(void)
{
       t_disco *vec_discos;
       int ntd;           /* número total de discos */
       char cantante[N];  /* nombre del cantante */
       t_disco pdc;       /* primer disco encontrado del cantante */
       int ndc;           /* número de discos del cantante */

       /*Asignación dinámica e Inicialización del vector*/
       ...

       /* Llamada a la función */
       ....................................................................

}
```

1.10 Bucles

Indicar las líneas que faltan para programar la suma de una serie cuyo término general es n/n!

```
#include <stdio.h>
int main(void)
{
       int i,j,n,num,suma,fact;

       n=5;
       .........................
       for (i=1;i<=n;i++){
              num=i;
              ...............................
              for(j=1;j<=n;j++)
              {
                     ...............................
              }
              suma=suma+num/fact;
       }
}
```

1.11 Bucles

¿Cuál es la salida del siguiente código?

```
int main( )
{
	int i,j,x;

	i=1;
	j=0;
	x=0;
	while( i < 5 || j > i+1)
	{
		switch(i + j – 2){
			case 0 :
				x += 3;
				break;
			case 1 :
				x += 4;
				break;
			default :
				x++;
				break;
		}
		printf(“%d”, x);
		++i;
		++j;
	}
}
```

Salida.....

1.12 Punteros

Considerando las siguientes declaraciones, escribir el valor de cada asignación:

```
...
char u,v;
char *pu,*pv;
...
v = ’n’;
pv = &v;
*pv = v + 3;
u = *pv + 5;
pu = &u;
...
```

Respuesta: `u = ........, v = ..........., *pu = ........, *pv = ..........`

1.13 Punteros

Escribir el código de la función Copiar que copia una cadena de caracteres en otra (equivalente a `strcpy`). El prototipo debe ser:

```
void Copiar(char *destino, char *origen);
```

1.14 Recursividad

Diseñar una función recursiva para calcular los números combinatorios como sigue:

$C(m, n) = 1$ si $n = 0$ ó si $n = m$

$C(m, n) = m$ si $n = 1$ ó si $n = m - 1$

Para cualquier otro caso:

$C(m, n) = C(m-1, n-1) + C(m-1, n)$

1.15 Vectores y Matrices

Escribir una **función** para calcular la traspuesta de una matriz tipo `float`. La función recibe 2 matrices, la matriz original y la matriz traspuesta. Se considera que la matriz original está completamente llena y tiene tamaño NxM, mientras que la matriz traspuesta tiene tamaño MxN y está sin inicializar.

1.16 Listas y recursividad

Suponiendo que tenemos una lista enlazada ya creada, ¿cuál sería la salida de la función `Mostrar`?

Código de la función `Mostrar`:

```
void Mostrar(t_persona *aux)
{
      if (aux!=NULL) {
         Mostrar(aux->siguiente);
         if (aux->edad >= 10) {
            printf("%s %s: %d años\n",
               aux->nombre,
               aux->apellidos,
               aux->edad);
         }
      }
}
```

La estructura t_persona está declarada de la siguiente manera:

```
typedef struct persona {
   char nombre[25];
   char apellidos[50];
   int edad;
   struct persona *siguiente;
} t_persona;
```

Suponer que los datos de la lista son:

Antonio, López, 23 → Juan, García, 9 → Andrés Aguilera, 5 → Antonio Sánchez, 10 → NULL

Salida de la función `Mostrar`:

1.17 Punteros

¿Cuál es la salida del siguiente código?

```
#include <stdio.h>
char func ( char *c1 , char *c2 )

int main( )
{
       char a = 's' , b = 'n' , c ;
       c = func( &a , &b ) ;
       printf("%c , %c , %c" , a , b , c ) ;
}

char func ( char *c1 , char *c2 )
{
   char c='z' ;
   *c1 = 'p' ;
   *c2 = 'q' ;
   if (*c1==*c2) {
      c=*c1;
   }
   return c;
}
```

Salida: .....

1.18 Vectores y Matrices

¿Cuál es la salida del siguiente código?

```
int main(void)
{
   int z[3][4] = {{1,2,3,4},{5,6,7,8},{9,10,11,12}};
   int c;

   c=calculo(z);
   printf("%d", c);
}

int calculo(int x[][4])
{
   int i, j, res;
   for(j = 0; j<4; ++j) {
       res = j;
       for(i = 0; i<3; ++i) {
              if(x[i][j] > res )  res = x[i][j];
       }
   }
   return res;
}
```

Salida: .....

1.19 Cadenas de caracteres

Indicar la salida por pantalla del siguiente código:

```
       ...
       int j=0 ;
       char cad[15];
       strcpy(cad,"meteorologia");
       while(j<7){
              if(*(cad+j)>'m' && *(cad+j)<'s')
                     putchar(cad[j]);
              ++j;
       }
       ...
```

Salida .....

1.20 Punteros

Indicar la salida por pantalla del siguiente código:

```
int main()
{
  char cadena[]= “programa en C”;
  char *pc;
  int i;
  for(i=0;i<8;++i) {
    putchar(cadena[i]);
  }
  i=i+4;
  pc=cadena+i;
  puts(pc);
  return 0;
}
```

Salida:.....

Página extra para preguntas cortas

Segunda Parte: Códigos (5 puntos)

Problema 1 (2 puntos)

Diseñar un programa para llevar el control de las vitaminas que tienen diversos alimentos. Se parte de un archivo binario que contiene los datos de los alimentos y que ha sido creado con la siguiente estructura:

```
typedef struct {
   char nombre[100];
   float A,B6,B12,C,D
} t_alimento;
```

El programa debe realizar las siguientes operaciones:

1. Leer el archivo binario guardando los datos en un vector de estructuras (vector estático). Función `Leer`
2. Rellenar una matriz con ceros y unos donde cada fila representa un alimento y cada columna una de las 5 vitaminas estudiadas. Se pone 1 donde el nivel de la vitamina es mayor que 0.1 y se pone 0 en caso contrario. Función `Rellenar`
3. Obtener los siguientes resultados globales:
   - 3.1. Para cada una de las 5 vitaminas, indicar qué porcentaje de los alimentos estudiados contienen esa vitamina. Función `MostrarPorcentajes`
   - 3.2. Mostrar los nombres de los alimentos que tengan el máximo número de vitaminas. Función `MostrarAlimentos`

Problema 2 (1,5 puntos)

Un profesor de matemáticas desea crear para los alumnos de su clase un programa que les ayude a determinar el seno de los ángulos. El programa consiste en un vector dinámico de estructuras de tipo `t_angulo`. Cada estructura de tipo `t_angulo` contiene los siguientes campos: Angulo en grados y Seno del ángulo. Para crear el programa utilizar los siguientes pasos:

1-Preguntar el ángulo inicial de la tabla, el ángulo final y el intervalo de ángulo (incremento). Todo en grados

2-Calcular el tamaño de la tabla y construir un vector asignando memoria dinámica al puntero de estructuras

3-Llamar a la función `Inicializar` que rellene los valores de los ángulos en el vector. En grados

4-Llamar a la función `Calcular` que rellene los valores del seno a partir de los valores de ángulos del vector. El seno del ángulo se calcula mediante la relación siguiente, para ángulos en radiantes:

$\text{Seno}(x) = x - x^3/3! + x^5/5! - x^7/7! + \ldots\ldots + (-1)^n (x^{2n+1})/(2n+1)!$

<u>Nota:</u>

Para calcular el seno de un ángulo se debe sumar los términos de la serie necesarios hasta obtener un término cuyo valor absoluto sea menor que 0.000001. Se supone que la función `Factorial` es conocida.

Se sabe que el paso de grados a radianes realiza: `radianes = (grados * M_PI)/180`.

Problema 3 (1,5 puntos)

Se tiene un archivo de texto (`nuevos.txt`) con el teléfono, la fecha de nacimiento y el nombre de varias personas que se quieren añadir a una base de datos (archivo binario `agenda.dat`). Se trata de leer del archivo de texto los datos de cada persona y añadirlos en el archivo binario.

No se puede utilizar vectores porque se trata de hacer todas las operaciones trabajando directamente sobre los archivos. La estructura del archivo binario es la siguiente:

```
typedef struct {
   int dia;
   int mes;
   int aNo;
} t_fecha;
typedef struct {
   long telefono;
   t_fecha fecha_nacimiento;
   char nombre[20];
} t_persona;
```