Programación en C/C++

Hoja de problemas 2 :

“Tipos de datos derivados”

David Rozas Domingo
I.T.I.Sistemas

Responda a las siguientes preguntas :

1) ¿Qué valores toman las variables n1 y n2 tras ejecutar el siguiente fragmento de código?

```
.....
float n1 = 10;
float n2 = 5;
float *p, *q;
.....
p = &n1;
q = &n2;
*q = *p + *q;
.....
```

```
Apuntamos a ambas variables. La ultima sentencia suma el valor apuntado
por p (10) a n2. Seria equivalente a hacer n2=n1+n2
El valor de n1 es 10, y el de n2 es 15.
```

2) Dadas las siguientes definiciones de variables:

```
int x;
int *p1;
int **p2;
```

¿Cuál de las siguientes sentencias permite que x tome el valor 4 de forma correcta?

```
/*sentencia a* : La asignacion del puntero es incompatible*/
p1 = &p2;
*p2 = &x;
*p1 = 4;
```

```
/*sentencia b : La asignacion del puntero es incompatible: crea un puntero
hacia un entero sin conversion*/
p2 = &x;
*p2=4;
```

```
 /*sentencia c*/
 p2 = p1;/*asignacion a un tipo de puntero incompatible*/
 p1 = &x;
 *p2 = 4;/*crea un puntero hacia un entero sin conversion*/
```

```
 /*sentencia d*/
 p2 = &p1;/*Apuntamos a la direccion de memoria del puntero p1*/
 p1 = &x;/*Apuntamos a la direccion de memoria de x*/
 **p2 = 4;/*Accedemos al valor, del valor del puntero*/
```

Por tanto, la sentencia correcta es la d.

3) Dado el siguiente programa escrito en C, explicar razonadamente que hace.

```
#include <stdio.h>
#include <stdlib.h>


int main(void)
{
	int **p;
	int n, m , i;

	/*Pide al usuario un numero mayor que 0*/
	do
	{
		printf("Valor de n : \n");
		scanf("%d", &n);
	}while(n<=0);


	/*Reservamos memoria para n punteros a enteros*/
	p = (int**)malloc(n*sizeof(int*));

	/*Para cada puntero a entero*/
	for(i=0; i<n; i++)
	{
		/*Pedimos un numero de elementos*/
		int j;
		printf("Numero de elementos de la componente %d", i+1);
		scanf("%d", &m);

		/*Y reservamos memoria para m enteros*/
		p[i] = (int *)malloc(m*sizeof(int));
		/*Y a continuacion pedimos dichos valores*/
		for (j=0; j<m; j++)
			scanf("%d", &p[i][j]);
	}


	return 1;
}
```

Como se va indicando a través de los comentarios que se han agregado, el objetivo del programa es crear una matriz dinámica de valores enteros,solicitando su tamaño y sus valores al usuario.

Desarrolle programas en C para los siguientes problemas propuestos.

Nota 1 : El compilador gcc da una advertencia indicando que main debe devolver algo. Por ello en todos los programas main devuelve un int (0 en caso de ejecución correcta).

Nota 2 : La librería <math.h> me ha dado problemas en el ejercicio 9 (no he podido usar la función sqrt). Por ello se prescindió de ella, pero se indica como sería su cálculo exacto, en un comentario.

4)

```
/************************************************************
       Asignatura : Programacion en C/C++
       Practica : Hoja de problemas 2
       Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 4. Calcula la suma de los elementos de una matriz
                                        que no forman parte del diagonal*/

#include <stdio.h>
#define N 4

int main(void)
{
       int i, j,suma;
       int matriz[N][N];

       /*Pedimos los valores al usuario*/
       for(i=0;i<N;i++)
              for(j=0;j<N;j++)
              {
                     printf("Introduce el valor matriz[%i][%i] :", i+1, j+1);
                     scanf("%i", &matriz[i][j]);
              }

       /*Pintamos la matriz por pantalla, y calculamos la suma de elementos*/
       for(i=0, suma=0;i<N;i++)
              for(j=0;j<N;j++)
              {
                     printf("%5i", matriz[i][j]);
                     if (j==(N-1))
                            printf("\n");
                     /*Sumamos todos los elementos menos los de la diagonal*/
                     if (i!=j)
                            suma  = suma + matriz[i][j];
              }

       printf("La suma de todos los elementos que no estan en la diagonal es : %i",
suma);

       return 0;
}
```

5)

```
/************************************************************
       Asignatura : Programacion en C/C++
       Practica : Hoja de problemas 2
       Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 5. Define una estructura de datos fecha.
                 Recibe dos fechas y las imprime en orden */

#include <stdio.h>

/*Definicion  y renombramiento de tipo fecha*/
struct tipoFecha{
       int dia,mes;
       unsigned anio;
       };
typedef struct tipoFecha tFecha;

int main(void)
{
       tFecha fecha1,fecha2;
       int fecha1mayor = 0;
       /*Pedimos dos fechas */
       do
       {
              printf("Introduzca el dia de la primera fecha : ");
              scanf("%i", &fecha1.dia);
       }while(fecha1.dia<1 || fecha1.dia>31);

       do
       {
              printf("Introduzca el mes de la primera fecha  :");
              scanf("%i", &fecha1.mes);
       }while(fecha1.mes<1 || fecha1.mes>12);

       printf("Introduzca el año de la primera fecha : ");
       scanf("%i", &fecha1.anio);


       do
       {
              printf("Introduzca el dia de la segunda fecha : ");
              scanf("%i", &fecha2.dia);
       }while(fecha2.dia<1 || fecha2.dia>31);

       do
       {
              printf("Introduzca el mes de la segunda fecha  :");
              scanf("%i", &fecha2.mes);
       }while(fecha2.mes<1 || fecha2.mes>12);

       printf("Introduzca el año de la segunda fecha : ");
       scanf("%i", &fecha2.anio);


       /*Y a continuacion procedemos a compararlas */

       if (fecha1.anio==fecha2.anio)
       {
              if(fecha1.mes==fecha2.mes)
              {
                     if(fecha1.dia==fecha2.dia)
                     {
                            fecha1mayor=0;
                     }else if(fecha1.dia>fecha2.dia){
                            fecha1mayor=1;

                     }else{
                            fecha1mayor=-1;
```

```
                }

        }else if(fecha1.mes>fecha2.mes){
                fecha1mayor=1;

        }else{
                fecha1mayor=-1;
        }

    }else if(fecha1.anio>fecha2.anio){
        fecha1mayor=1;

    }else{
        fecha1mayor=-1;
    }

    /*Y mostramos el resultado por pantalla*/
    switch(fecha1mayor){
        case 0 :
            printf("¡Las dos fechas son iguales!\n");
            printf("\t%2i-%i-%i\n", fecha1.dia, fecha1.mes, fecha1.anio);
            printf("\t%2i-%i-%i\n", fecha2.dia, fecha2.mes, fecha2.anio);
            break;
        case 1 :
            printf("La fecha 1 es mayor :\n");
            printf("\t%2i-%i-%i\n", fecha1.dia, fecha1.mes, fecha1.anio);
            printf("\t%2i-%i-%i\n", fecha2.dia, fecha2.mes, fecha2.anio);
            break;
        case -1 :
            printf("La fecha 2 es mayor :\n");
            printf("\t%2i-%i-%i\n", fecha2.dia, fecha2.mes, fecha2.anio);
            printf("\t%2i-%i-%i\n", fecha1.dia, fecha1.mes, fecha1.anio);
    }
    return 0;
}
```

6)

```
/************************************************************
    Asignatura : Programacion en C/C++
    Practica : Hoja de problemas 2
    Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 6. Define una estructura de datos fecha.
                 Recibe dos fechas y las imprime en orden, pero
                          accediendo por punteros*/

#include <stdio.h>
#include <stdlib.h>

/*Definicion  y renombramiento de tipo fecha*/
struct tipoFecha{
    int dia,mes;
    unsigned anio;
    };
typedef struct tipoFecha tFecha;

int main(void)
{
    tFecha *ptFecha1, *ptFecha2;
    int fecha1mayor = 0;
    /*Reservamos memoria para ambas fechas con cada puntero*/
    ptFecha1 = (tFecha*)malloc(sizeof(tFecha));
    ptFecha2 = (tFecha*)malloc(sizeof(tFecha));

    if ((ptFecha1 != NULL) && (ptFecha2 != NULL))
    {
        /*Pedimos dos fechas */
        do
```

```
		{
			printf("Introduzca el dia de la primera fecha : ");
			scanf("%i", &ptFecha1->dia);
		}while(ptFecha1->dia<1 || ptFecha1->dia>31);

		do
		{
			printf("Introduzca el mes de la primera fecha  :");
			scanf("%i", &ptFecha1->mes);
		}while(ptFecha1->mes<1 || ptFecha1->mes>12);

		printf("Introduzca el año de la primera fecha : ");
		scanf("%i", &ptFecha1->anio);


		do
		{
			printf("Introduzca el dia de la segunda fecha : ");
			scanf("%i", &ptFecha2->dia);
		}while(ptFecha2->dia<1 || ptFecha2->dia>31);

		do
		{
			printf("Introduzca el mes de la segunda fecha  :");
			scanf("%i", &ptFecha2->mes);
		}while(ptFecha2->mes<1 || ptFecha2->mes>12);

		printf("Introduzca el año de la segunda fecha : ");
		scanf("%i", &ptFecha2->anio);


		/*Y a continuacion procedemos a compararlas */
		if (ptFecha1->anio == ptFecha2->anio)
		{
			if(ptFecha1->mes == ptFecha2->mes)
			{
				if(ptFecha1->dia == ptFecha2->dia)
				{
					fecha1mayor=0;
				}else if(ptFecha1->dia > ptFecha2->dia){
					fecha1mayor=1;

				}else{
					fecha1mayor=-1;
				}

			}else if(ptFecha1->mes > ptFecha2->mes){
				fecha1mayor=1;

			}else{
				fecha1mayor=-1;
			}

		}else if(ptFecha1->anio > ptFecha2->anio){
			fecha1mayor=1;

		}else{
			fecha1mayor=-1;
		}

		/*Y mostramos el resultado por pantalla*/
		switch(fecha1mayor){
			case 0 :
				printf("¡Las dos fechas son iguales!\n");
				printf("\t%2i-%i-%i\n", ptFecha1->dia, ptFecha1->mes,
ptFecha1->anio);
				printf("\t%2i-%i-%i\n", ptFecha2->dia, ptFecha2->mes,
ptFecha2->anio);
				break;
			case 1 :
				printf("La fecha 1 es mayor :\n");
```

```
                        printf("\t%2i-%i-%i\n", ptFecha1->dia, ptFecha1->mes,
ptFecha1->anio);
                        printf("\t%2i-%i-%i\n", ptFecha2->dia, ptFecha2->mes,
ptFecha2->anio);
                        break;
                  case -1 :
                        printf("La fecha 2 es mayor :\n");
                        printf("\t%2i-%i-%i\n", ptFecha2->dia, ptFecha2->mes,
ptFecha2->anio);
                        printf("\t%2i-%i-%i\n", ptFecha1->dia, ptFecha1->mes,
ptFecha1->anio);
            }

            /*Por ultimo, liberamos la memoria*/
            free(ptFecha1);
            free(ptFecha2);
            return 0;

      }else{
            printf("Error : No hay espacio en memoria suficiente\n");
            return -1;
      }
}
```

7)

```
/************************************************************
      Asignatura : Programacion en C/C++
      Practica : Hoja de problemas 2
      Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 7. Calcula la ecuacion de la recta, dados
                 dos de sus puntos*/

#include <stdio.h>

/*Definicion  y renombramiento de tipo punto*/
struct tipoPunto{
      float x,y;
      };
typedef struct tipoPunto tPunto;

int main(void)
{
      tPunto pto1,pto2;
      float m,b;


      /*Obtenemos los valores de los puntos*/
      printf("Vamos a calcular la ecuacion de una recta, dados dos puntos.\n");
      printf("\t Primer punto \n");
      printf("\t ¿Valor de x? : ");
      scanf("%f", &pto1.x);
      printf("\t ¿Valor de y? : ");
      scanf("%f", &pto1.y);
      printf("\t Segundo punto \n");
      printf("\t ¿Valor de x? : ");
      scanf("%f", &pto1.x);
      printf("\t ¿Valor de y? : ");
      scanf("%f", &pto1.y);

      /*Calculamos pendiente e interseccion*/
      m = (pto2.y-pto1.y)/(pto2.x-pto1.x);
      b = pto1.y - m*pto1.x;

      printf("\n ### La ecuacion de la recta es y = %.2f*x + %.2f ###\n", m, b);
      return 0;
}
```

8)

```
/************************************************************
       Asignatura : Programacion en C/C++
       Practica : Hoja de problemas 2
       Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 8. Busca los puntos de silla de una matriz*/

#include <stdio.h>
#define N 4

struct tipoPosicion{
       int fila;
       int columna;
       };
typedef struct tipoPosicion tPosicion;

int main(void)
{
       int i, j, k, val_aux, alguno;
       int matriz[N][N];
       tPosicion min_fila[N]; /*Guarda las posiciones de los minimos de cada fila*/
       tPosicion max_columna[N]; /*Guarda las posiciones de los maximos de cada columna*/

       for(i=0; i<N; i++)
              for(j=0; j<N; j++)
              {
                     printf("Introduce el valor [%i,%i]de la matriz : ", i,j);
                     scanf("%i", &matriz[i][j]);
              }

       printf("\n\n");
       /*Pintamos la matriz*/
       for(i=0; i<N; i++)
       {
              printf("|");
              for(j=0; j<N; j++)
              {
                     printf("  %2.i  ", matriz[i][j]);
              }
              printf("|\n");
       }

       /*Obtenemos un vector con las posiciones minimas de cada fila*/
       k = 0;
       for(i=0; i<N; i++)
       {
              val_aux = matriz[i][0];
              min_fila[k].columna = 0;
              min_fila[k].fila = i;
              for (j=0; j<N; j++)
              {
                     if (matriz[i][j]<val_aux)
                     {
                            val_aux = matriz[i][j];
                            min_fila[k].columna = j;
                     }
              }

              k++;
       }




       /*Obtenemos un vector con las posiciones maximas de cada columna*/
       k = 0;
       for(i=0; i<N; i++)
       {
              val_aux = matriz[0][i];
```

```
			max_columna[k].fila = 0;
			max_columna[k].columna = i;
			for(j=0; j<N; j++)
			{
				if(matriz[j][i]>val_aux)
				{
					val_aux = matriz[j][i];
					max_columna[k].fila = j;
				}
			}
			k++;
		}


		/*Por último, comparamos las posiciones obtenidas en ambos vectores*/
		alguno = -1;
		for(i=0; i<N; i++)
		{
			for(j=0; j<N; j++)
			{
				if
((max_columna[i].fila==min_fila[j].fila)&&(max_columna[i].columna==min_fila[j].columna))
				{
					printf("\nHay un punto de silla en : [%i , %i] \n",
max_columna[i].fila, max_columna[i].columna);
					alguno = 1;
				}
			}
		}

		if (alguno<0)
			printf("No hay puntos de silla\n");

		return 0;
}
```

9)

```
/*************************************************************
	Asignatura : Programacion en C/C++
	Practica : Hoja de problemas 2
	Autor : David Rozas Domingo
*************************************************************/

/* Descripcion : Ejercicio 9. Calcula los puntos mas cercanos*/
/*PROBLEMA : problema con libreria math.h*/

#include <stdio.h>
#include <math.h>
#define N 4


/*Creamos una estructura de tipoPunto*/
struct tPunto{
	float x1;
	float x2;
};
typedef struct tPunto punto;

/*Declaracion de funciones auxiliares*/
float distanciaEuclidea(punto x, punto y);


int main(void)
{
	punto puntos[N];

	int i,j,p1,p2;
	float aux, actual;

	/*Leemos las coordenadas de los puntos que nos da el usuario*/
```

```
	for(i=0; i<N; i++)
	{
		printf("Introduce la coordenada x el %i° punto : ", i+1);
		scanf("%f", &puntos[i].x1);
		printf("Introduce la coordenada y el %i° punto : ", i+1);
		scanf("%f", &puntos[i].x2);
	}

	/*Mostramos la informacion */
	for(i=0; i<N; i++)
	{
		printf("Pto %i : (%.2f,%.2f)\n", i+1, puntos[i].x1, puntos[i].x2);
	}

	/*A continuacion calculamos las distancias euclídeas de todos con todos
	y nos guardaremos las referencias de la menor*/

	i = 0;
	j = 0;
	p1 = 0;
	p2 = 0;
	/*Inicializamos con el primer par de puntos*/
	actual = distanciaEuclidea(puntos[0],puntos[1]);
	aux = 0.0;

	while(i<N)
	{
		j = i + 1;
		while(j<N)
		{
			aux = distanciaEuclidea(puntos[i],puntos[j]);
			printf("Distancia entre  (%.2f , %.2f) con (%.2f , %.2f) = ",
puntos[i].x1, puntos[i].x2, puntos[j].x1, puntos[j].x2);
			printf("%.2f\n", aux);
			if (aux<actual)
			{
				actual = aux;
				p1 = i;
				p2 = j;
			}
			j++;
		}
		i++;
	}


printf("\n---------------------------------------------------------------------------------
\n");
	printf("Los puntos mas cercanos son (%.2f , %.2f) con (%.2f , %.2f)",
puntos[p1].x1, puntos[p1].x2, puntos[p2].x1, puntos[p2].x2);

printf("\n---------------------------------------------------------------------------------
\n");

	return 0;
}

/*************************************************************/
/*                    Funciones auxiliares                   */
/*************************************************************/


float distanciaEuclidea(punto x, punto y)
{
	/*La distancia euclídea en R2 es:
	d((x1, x2),((y1,y2))=((x1-y1)^2+(x2-y2)^2)^1/2*/

	/*POR PROBLEMAS CON LIBRERIA math.h NO SE CALCULA CORRECTAMENTE.
	  EN REALIDAD EL RESULTADO CORRECTO SE CALCULA :
	float res = sqrt(((x.x1 - y.x1)*(x.x1 - y.x1)) + ((x.x2 - y.x2)*(x.x2 - y.x2)));*/
```

```
	float res = ((x.x1 - y.x1)*(x.x1 - y.x1)) + ((x.x2 - y.x2)*(x.x2 - y.x2));
	return res;
}
```

10)

```
/************************************************************
	Asignatura : Programacion en C/C++
	Practica : Hoja de problemas 2
	Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 10. Comprueba si un vector de enteros
                          esta contenido en otro*/

#include <stdio.h>
#define N 4
#define M 5


struct tipoMarca{
	int data;
	int marcado;
	};
typedef struct tipoMarca tMarca;


int main(void)
{
	int v1[N];
	tMarca v2[M];
	int i, encontrado, j;

	/*Pedimos los valores al usuario, e inicializamos arrays*/
	for(i=0;i<N;i++)
	{
		printf("Introduce el valor v1[%i] : ", i+1);
		scanf("%i", &v1[i]);
	}
	for(i=0;i<M;i++)
	{
		printf("Introduce el valor v2[%i] : ", i+1);
		scanf("%i", &v2[i].data);
		v2[i].marcado = -1;
	}

	/*Recorremos el array origen comparando con el destino. Usamos marcas
	para el array destino para no falsear en caso de elementos repetidos :
	Ej.: [1,1,1] no esta contenido en [1,2,3,4]*/

	i=0;
	encontrado = 1;
	while ((i<N)&&(encontrado>0))
	{
		encontrado = -1;
		j = 0;
		while((j<M)&&(encontrado<0))
		{
			if ( (v1[i]==v2[j].data)&&(v2[j].marcado<0) )
			{
				/*Marcamos y forzamos salida del bucle interno y
				  permanencia en bucle externo*/
				v2[j].marcado = 1;
				encontrado = 1;
			}

			j++;
		}

		i++;
```

```
	}

	printf("\n----------------------------------------\n");

	if (encontrado>0)
		printf("El vector v1 esta contenido en v2\n");
	else
		printf("El vector 1 no esta contenido en v2\n");

	printf("----------------------------------------\n");
}
```

11)

```
/************************************************************
	Asignatura : Programacion en C/C++
	Practica : Hoja de problemas 2
	Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 11. Creacion de lista enlazada, y recorrido
                 para obtener la suma de sus numeros*/

#include <stdio.h>
#include <stdlib.h>

/*Definiciones para tipo lista*/
struct tipoNodo{
	float info;
	struct tipoNodo *sig;
	};
typedef struct tipoNodo tNodo;


int main(void)
{
	tNodo *lista, *pAux, *indice;
	int n_elementos, i;
	float valorAux, total;

	lista = NULL;
	pAux = NULL;
	indice = NULL;
	total = 0.0;

	do
	{
		printf("¿Cuantos elementos quieres insertar? : ");
		scanf("%i", &n_elementos);
	}while(n_elementos<1);


	for(i=0; i<n_elementos; i++)
	{
		printf("Inserta un valor : ");
		scanf("%f", &valorAux);

		/*Reservamos memoria, e introducimos informacion*/
		pAux = (tNodo*)malloc(sizeof(tNodo));
		pAux->info = valorAux;

		/*Insercion, dependiendo de si es o no el primer elemento*/
		if (lista==NULL)
		{
			pAux->sig = NULL;
			lista = pAux;
		}else{
			pAux->sig = lista;
			lista = pAux;
		}
	}
```

```
	/*Recorremos la lista, y hacemos el sumatorio*/
	indice = lista;
	while(indice != NULL)
	{
		total = total + indice->info;
		indice = indice->sig;
	}

	printf("\n-----------------------------------------------------------\n");
	printf("Suma total de los elementos de la lista : %.2f", total);
	printf("\n-----------------------------------------------------------\n");

	/*Por ultimo, liberamos la memoria*/
	while(lista != NULL)
	{
		indice = lista;
		lista = lista->sig;
		free(indice);
	}

	return 0;
}
```

12)

```
/************************************************************
	Asignatura : Programacion en C/C++
	Practica : Hoja de problemas 2
	Autor : David Rozas Domingo
************************************************************/

/* Descripcion : Ejercicio 12. Ordena un vector de enteros mediante
                          algoritmo de insercion directa*/

#include <stdio.h>
#define N 5

int main(void)
{
	int v[N];
	int i,j,aux;


	for(i=0; i<N; i++)
	{
		printf("Inserta un elemento : ");
		scanf("%i",&v[i]);
	}

	printf("Vector sin ordenar : \n [ ");
	for(i=0; i<N; i++)
		printf(" %i ", v[i]);

	printf("]\n");


	/*Cada elemento del vector se compara en cada iteracion con los que
	le preceden, "hasta que encontramos su sitio"*/
	for(i=1; i<N; i++)
	{
		aux = v[i];
		j = i - 1;
		/*Desplazamos todos los elementos mayores que v[i]*/
		while( (v[j]>aux) && (j>=0))
		{
			printf("Comparando %i con %i\n", v[j],aux);
			v[j+1] = v[j];
			j--;
		}
```

```
			/*Copiamos v[i] a su pos final*/
			v[j+1] = aux;
		}


		printf("Vector ordenado : \n [ ");
		for(i=0; i<N; i++)
			printf(" %i ", v[i]);

		printf("]\n");

		return 0;
}
```