Hoja de ejercicios resueltos Tema 1

• *Operadores*

■ *Ejercicio 1.- Descubrir errores*

```
#include <stdio.h>
void main(void){
    int i = 0;
    while(i<100)
    {
         printf(“%d\n”, i);
         i++;
         if(i=50)
              break;
    }
    printf(“Acabado el bucle while”);
 }
```

- *Solución ejercicio 1*

Dentro del if, ponemos i=50, cuando debemos poner i==50. Qué pasa cuando ponemos i=50? Se asigna 50 a i, se evalúa i. Cómo i es mayor que cero, la expresión del if resulta ser cierta, con lo que se ejecuta break y sale del bucle.

■ *Ejercicio 2.- Descubrir errores*

```
#include <stdio.h>
void main(void){
    int i = 0;
    while(i<10)
    {
         if(i=!0)
              printf("%d\n", i);
         i++;
    }
    printf("Acabado el bucle while");
}
```

- *Solución ejercicio 2*

  Esperamos que imprima todos los números del 0 al 10 menos el 0. Pero resulta que imprime muchos 1. Dentro del if, otra vez. Estamos negando el cero, asignándolo a i y ejecutando la sentencia dentro del if.

• *Punteros*

■ *Ejercicio 3: Comentar el siguiente programa*

```
int a, b; //Variables de tipo entero
int *pa; //Puntero a un entero

a = 5;
pa = &a; // pa apunta a la variable a.
b = *pa; // b vale 5
```

■ *Ejercicio 4: Comentar el siguiente programa*

```
int i,j,*p;

p=&i;  // p tiene la dirección de i
*p=21; // al cambiar el valor del puntero que apunta a la dirección
de i, cambiamos también el valor de i a 21;
p=&j;  // ahora p tendrá la dirección de j en lugar de la de i
*p=1;  // j tendrá el valor 1
```

■ *Ejercicio 5: Casting y punteros a void: Comentar el siguiente programa e identificar errores*

```
int *p;          // Definimos un puntero a entero
double *q;       // Definimos un puntero a un double
void *r;         // Definimos un puntero a void a través del
                 cual podemos asignarle cualquier tipo de
                 puntero
p=q;             // ERROR: NO se puede pq apuntan a distintos
                 tipo de variable
p=(int *)q;      // SI se puede pq hacemos casting
p=r=q            // SI se puede pq usamos antes r
```

- *Ejericio 6: Aritmética de punteros: Comentar el siguiente programa*

```
void main(void){

   int a,b,c;     // Declaramos 3 enteros
   int *p1,*p2;   // Declaramos 2 punteros

   p1 = &a;        // p1 apunta a la dirección de a
   *p1 = 1;        // modificamos el valor de p1 y por tanto a=1
   p2 = &b;        // p2 apunta a la dirección de b
   *p2 = 2;        // modificamos el valor de p2 y por tanto b=2
   p1 = p2;        // le asignamos el mismo valor de dirección de p2 a
                   p1, por tanto p1 apuntará ahora a b
   *p1 = 0;        // por tanto b = 0
   p2 = &c;        // p2 apunta a la dirección de c
   *p2 = 3;        // el valor de p2 será 3, por tanto c=3
}
```

- ***Ejercicio propuesto 1 para resolver en clase: Aritmética de punteros***

```
#include <stdio.h>

void main( void )
{
int u = 3, v;
int *pu; //puntero a entero
int *pv; //puntero a entero

pu = &u;
v = *pu;
pv = &v;
printf("\nu=%d &u=%X  pu=%X *pu = %d", u, &u, pu, *pu);
printf("\nv=%d &v=%X  pv=%X *pv = %d", v, &v, pv, *pv);
}
```

  - *Solución ejercicio propuesto 1*

Salida:

u = 3	**&u = ff56**	**pu = ff56**	***pu = 3**
v = 3	**&v = f67A**	**pv = f67A**	***pv = 3**

- ***Ejercicios propuestos 2 para resolver en clase: Aritmética de punteros***

```
#include <stdio.h>

void main(void){
     int a=5,b,*pa,*pb;

     printf("La direccion de a= %X\n",&a);
     printf("El valor de a= %X\n",a);
     pa=&a; // puntero apunta a la direccion de a
     printf("La dirección del puntero pa es %X\n",&pa);
     printf("El puntero pa apunta a la dirección %X\n",pa);
     printf("El valor de la dirección de memoria a la que apunta el puntero pa
es %d\n\n",*pa);
     pb=pa; // el puntero pb apunta a la direccion del puntero pa
     printf("La dirección del puntero pb es %X\n",&pb);
     printf("El puntero pb apunta a la dirección %X\n",pb);
      printf("El valor de la dirección de memoria a la que apunta el puntero pb
es %d\n",*pb);

     system("PAUSE");
}
```

- *Solución ejercicio propuesto 2*

  -La dirección de a = 22FF74
  -El valor de a = 5
  -La dirección del puntero pa es 22FF6C
  -El puntero pa apunta a la dirección 22FF74
  -El valor de la dirección de memoria a la que apunta el puntero pa es 5

  -La dirección del puntero pb es 22FF68
  -El puntero pb apunta a la dirección 22FF74
  -El valor de la dirección de memoria a la que apunta el puntero pb es 5

- **Aritmética de punteros**

  - *Ejercicio 7: Aritmética de punteros: Comentar el siguiente programa*

```
char* ptrchar;
double* ptrdouble;
...
*(ptrchar+3) = 33; /* la dirección es ptrchar + 3 bytes */

*(ptrdouble+3) = 33.0; /* la dirección es ptrdouble + 24 bytes, ya que cada
double ocupa 8 bytes */
```

  - *Ejercicio 8: Aritmética de punteros con vectores: Comentar el siguiente programa*

```
int v[5];    // un vector de 5 elementos enteros
int *p;      // puntero a entero
...
p = &v[0];   /* ptr apunta al principio del vector */
*p = 1;      /* igual que vector[0] = 1 */
*(p+1) = 2;  /* igual que vector[1] = 2 */
*(p+2) = 3;  /* igual que vector[2] = 3 */
```

- **Funciones: Paso por dirección**

  - *Ejericio 1: Escribir la salida por pantalla del siguiente programa*

```
#include <stdio.h>
void func1(int u, int v);
void func2(int *pu, int *pv);
main ()
{
 int u=1;
 int v=3;
 printf("Antes de func1: u=%d, v=%d\n", u, v);
 func1(u, v);
 printf("Despues de func1: u=%d, v=%d\n", u, v);
 printf("Antes de func2: u=%d, v=%d\n", u, v);
 func2(&u, &v);
 printf("Despues de func2: u=%d, v=%d\n", u, v);
 system("PAUSE");
}

void func1(int u, int v)
{
 u=0;
 v=0;
 printf("Dentro de func2: u=%d, v=%d\n", u, v);
}
void func2(int *pu, int *pv)
{
 *pu=0;
 *pv=0;
 printf("Dentro de func2: *pu=%d, *pv=%d\n", *pu, *pv);
}
```

    - *Solución ejercicio 1*

      Antes de func1: u=1, v=3
      Dentro de func1: u=0, v=0
      Después de func1: u=1, v=3
      Antes de func2: u=1, v=3
      Dentro de func2: *pu=0,*pv=0
      Después de func2: u=0, v=0

- **Arrays y punteros**

  - *Ejercicio 1: Arrays y punteros: Escribir la salida por pantalla del siguiente programa*

```
#include <stdio.h>

void main(void){

int x[3]; // array de 3 enteros
int *puntero;
x[0]=10;
x[1]=20;
x[2]=30;
puntero = x;      // metodo 1
puntero = &x[0];   // metodo 2 ambos métodos son equivalentes
printf("%d\n\n",puntero[0]); // Mostramos el elemento 0 del array
printf("%d\n\n",*puntero); // Mostramos el elemento 0 del array
printf("%X\n\n",&puntero);  // Mostramos la posición en memoria del primer
elemento del array
printf("%X\n\n",&puntero[1]); // Mostramos la posición en memoria del segundo
elemento del array
printf("%d\n\n",puntero[1]);// Mostramos el segundo elemento del array
printf("%d\n",*(puntero+1)); // Mostramos el segundo elemento del array
printf("%d\n",*(puntero+2)); // Mostramos el tercer elemento del array
system("pause");

}
```

    - *Solución ejercicio 1*

```
Salida por pantalla
10
10
22FF5C
22FF64
20
20
30
```