Práctica 2 – Lenguaje C

Contenidos

- **Funciones y Macros.**
- **Pasaje de parámetros. Manejo de punteros**
- **Arreglos**

Práctico

1. Analice las siguientes funciones, ¿Qué cálculo realizan? ¿son equivalentes?

a)
```
int imin(int n, int m) {
  int min;
  if(n < m)
     min = n;
  else
     min = m;
  return min;
}
```

b)
```
int imin2(int n, int m) {
   if(n < m)
      return n;
   else
      return m;
}
```

c)
```
int imin3(int n, int m) {
  return (n < m)?n:m;
}
```

2. Describa diferencias entre las definiciones anteriores y la que sigue:

```
#define IMIN(n,m)    ((n) < (m) ? (n) :  (m))
```

3. Defina macros para las siguientes operaciones:

- ✓ obtener el valor absoluto de un número
- ✓ determinar si un caracter es dígito
- ✓ obtener el cubo de un número

4. ¿Por qué se dice que la elección entre una macro y una función es una "negociación" entre espacio de memoria y velocidad. Mencione una situación donde sea conveniente su uso.

5. Indique qué escribe el siguiente programa:

```
#include <stdio.h>
void escribe(int m);
int main(void) {
  int veces = 5;
  char ch = '!';   /* código  ASCII 33 */
  float f = 6.0;
  escribe(veces);
  escribe(ch);
  escribe((int)f);  /* conversión explicita*/
  return 0;
}
void escribe(int m) {
  while(m--)
     printf("#");
   printf("\n");
}
```

6. Dada la función *cambio* (para intercambio de dos enteros en memoria), describir detalladamente el comportamiento del programa para cada caso:

```
void cambio(int *da,
            int *db) {
 int aux;
 aux = *da ;
 *da = *db ;
 *db = aux ;
}
```

a)
```
void main() {
 int a = 5, b = 6;
 cambio(&a, &b);
}
```

b)
```
void main() {
 int a = 5, b = 6;
 cambio(a, b);
}
```

c)
```
void main() {
 int a = 5, b = 6;
 int *p = &a, *q = &b;
 cambio(p, q);
}
```

7. El siguiente programa lee un arreglo y lo modifica eliminando el elemento de la posición central. Se sabe que se han cometido algunos errores en los parámetros y punteros (direcciones de). Corregirlo para que funcione como se espera

```
#include <stdio.h>
void leevector(int a[], int *dn);
void elimina_central(int a, int n);
void escvector(int n[], int a);

int main(void) {               /* programa principal */
  int n;
  int a[20];
  leevector(&a, &n);
  elimina_central(a, *n);
  escvector(a[20], &n);
  return 0;
}

void leevector(int a[], int *dn) {
  int i;
  printf("Ingrese la cantidad  de elementos del vector<=100: ");
  scanf("%d", *dn);
  for(i=0; i<*dn; i++) {
     printf("Ingrese el elemento %d: ", i+1);
     scanf("%d", a[i]);
  }
}

void elimina_central (int a, int n) {
  int i;
  for(i = (n + 1) / 2; i < n-1; i++)
    a[i] = a[i+1];
  n--;
}

void escvector(int  n[], int a) {
   int i = 0;
   for(  ; i < a; i++)
     printf("%d  ", n[i]);
}
```

8. Indicar cuál es el efecto de las siguientes declaraciones:

a)
```
int a[20]
```

b)
```
int a[20], b[5]
```

c)
```
int n[5] = {10, 20, 30, 40, 50}
```

d)
```
float m[] = {4.7, 3.9, 2.0, 1.2, 0.6}
```

e)
```
char c[4] = {‘ h’, ‘o’, ‘l’, ‘a’}
```

f)
```
int d[6] = {1, 2, 3}
```

g)
```
float *pf, f[10];
pf = f;
```

h)
```
int a[5], b[5] = {0}
```

9. A partir de la declaración b) del ejercicio anterior, explicar si las siguientes sentencias son válidas. Justificar.

```
a  = b;
a[5] = b[5];
a = {1, 2, 3, 4, 5};
a[5] = {1, 2, 3, 4, 5};
```

10. Indicar el contenido de las variables luego de ejecutar las siguientes expresiones:

```
int *pint, y, var = 10;
pint = &var;
y = var = (*pint)*2;
```

11. ¿Que imprimen los siguientes algoritmos?

a)
```
void punt1() {
   int x, *p, **q;
   x = 10;
   p = &x;
   q = &p;
   printf("\n%d", **q);
}
```

b)
```
void punt2() {
   int x, *p, **q;
   x = 15;
   p = &x;
   q = &p;
   printf("\n%d ", **q***q);
   printf("%d ", *p**p);
   printf("%d ", **q**p);
}
```

c)
```
void punt3() {
   int x, *p, **q, y;
   x = 6;
   p = &x;
   q = &p;
   printf("\n%d ", **q + x - *p);
   y = 4; p = &y;
   printf("%d", **q + x - *p);
}
```

d)
```
void punt4() {
   int x, *p, **q, y;
   x = 10;
   y = x;
   p = &x;
   *p *= *p;
   q = &p;
   printf("\n%d", **q + *p + y/2);
}
```

12. Preguntas teóricas

a. ¿Qué diferencias existen entre una función y una MACRO?

b. ¿Cómo funcionan las MACRO?

c. ¿Qué tipo de pasaje de parámetros existe en C?

d. ¿Cómo se logra la comunicación bidireccional de un parámetro?

e. ¿Qué especifica que una función sea de tipo void?

f. ¿Por qué las funciones MACRO no pueden llevar ciclos?

g. Si se quiere implementar comunicación bidireccional de un puntero a int ¿Cómo lo haría?

h. ¿Cuál es el operador de indireccionamiento y cuál el de direccionamiento?

Ejercicios Adicionales

13. Analice el siguiente fragmento de código, especificando la salida del mismo al ser ejecutado.

```
#include <stdio.h>
int main () {
    int a = 1, b = 0;
    int* p = &a;
    printf ("a = %d ", a);
    printf ("p = %p ", p);
    printf ("*p = %d ", *p);
    a = 2;
    printf ("\na = %d ", a);
    printf ("p = %p ", p);
    printf ("*p = %d ", *p);
    *p *= 2;
    printf ("\na = %d ", a);
    printf ("p = %p ", p);
    printf ("*p = %d ", *p);
    p = &b;
    printf ("\na = %d ", a);
    printf ("p = %p ", p);
    printf ("*p = %d ", *p);
    return 0;
}
```

14. Desarrollar una función que utilice cambio para intercambiar de un arreglo de caracteres, los elementos simétricos (equidistantes al punto medio).

15. Explique cuáles son los errores de los siguientes trozos de código:

a)
```
int a[3] = {0, 1, 2, 3};
```

b)
```
int numero = 5;
 int* p_numero = numero;
```

c)
```
int a, b;
printf (“\nIngrese un numero: “);
sacnf(%d”,& a);
printf (“\n Ingrese otro número: “);
scanf (“%d”,&b);
(a = b)? printf(“Iguales”) :
         printf(“Distintos”);
```

d)
```
int* func () {
   int i = 5;
   return &i;
}
void main () {
   int* p_i = func();
   printf("%d", *p_i);
}
```

16. ¿Qué imprime el siguiente algoritmo?

```
int main () {
    int *ptrl, *ptr2;
    int a, b;
    { ptrl = &a;
    ptr2 = &b;
    *ptrl = 8;
    *ptr2 = 61;
    ptrl = ptr2;
    *ptrl +=  2;
    (*ptrl)++;
    printf("%d , %d \n", a, b);
    printf("%d, %d \n", *ptrl, *ptr2);
    printf("%p, %p \n", ptrl, ptr2);
    printf("%p, %p \n", &a, &b);
    }
    return 0;
}
```