Actividad 3

Apuntadores y Memoria Dinámica

Parte 1:

1. Dadas las siguientes declaraciones en C:

```
float x, *px, a[5];
```

¿Cuáles de las siguientes asignaciones son correctas y cuales son incorrectas? En el caso de las correctas explique qué hacen y para las incorrectas justifique su respuesta.

```
x = *px;
*px = x;
px = &x;
&x = px;
&(x+1) = x;
&(x)+1 = x;
*(&(x+1)) = x;
*(&(x)+1) = x;
x = a;
x = a[0];
x = *(a[1]);
x = (*a)[2];
x = a[3+1];
x = a[3]+1;
x = &((a[3])+1);
x = &(a[3])+1;
x = *(&(a[3])+1);
px = a;
px = a[0];
px = &(a[4]);
```

2. Dado el siguiente programa en C:

```
#include <stdio.h>

char string1[12] = {'H','o','l','a',' ','M','u','n','d','o','\0'};
char string2[] = "Hola Mundo";
char *string3 = "Hola Mundo";
char *ap_string[] = {"Uno","Dos","Tres"};
char abc[][6] = {"Uno","Dos","Tres"};

int main(void){
      printf("Tamaño de string1:%d\n",sizeof(string1));
      printf("Tamaño de string2:%d\n",sizeof(string2));
      printf("Tamaño de string3:%d\n",sizeof(string3));
      printf("Tamaño de ap_string:%d\n",sizeof(ap_string));
      printf("Tamaño de ABC:%d\n\n",sizeof(abc));
}
```

Se desea que usted lo compile y explique la salida.

3. Una cadena de caracteres (string) es un vector de caracteres que termina con un carácter '\0', por lo que las funciones que manipulan strings esperan que los mismos terminen en '\0'.

a. Dada la siguiente función

```
void strcat (char s[], char t[]) {
   int i, j;
   i = j = 0;
   while (s[i] != '\0') {
        i++;
   }
   while ((s[i++] = t[j++]) != '\0') {
   }
}
```

Reescríbala usando apuntadores. ¿Qué ventaja se tiene?

b. Explique qué hacen las siguientes funciones:

```
int my_strlen(const char *s){
   register const char *p = s;
   while(*p) //*p!='\0'
       p++;
   return p-s;
}

void my_strcpy(char *d, const char *f){
   while(*d++ = *f++);
}

void my_strcat(char *d, const char *f){
   my_strcpy(d+my_strlen(d),f);
}

char *my_strdump(const char *f){
   char *d;
   d=(char *)malloc(my_strlen(f)+1);
   my_strcpy(d,f);
   return d;
}

int my_substr(const char *s1, const char *s2){
   int i,v;
   const char *p1,*p2;
   v=my_strlen(s2)-my_strlen(s1);
   for(i=0;i<=v;i++)
       for(p1=s1,p2=s2+i;*p1++==*p2++;)
           if(*p1=='\0') return i;
   return -1;
}
```

c. Implemente, usando Lenguaje C, las siguientes funciones:

i. Que retorne el número de veces que el string s1 está en el

string s2.

```
int my_nsubstr(const char *s1, const char *s2)
```

ii. Invertir un string, por ejemplo, convierte a "abc" en "cba", trabaja sobre el mismo espacio apuntado por s.

```
void my_strrev(char *s)
```

4. Dado un arreglo de enteros, se desea que usted:

a. Haga una función que calcule la ocurrencia de un número dado.

b. Siguiendo el prototipo: **float relacionPI(int array[], int size)**, calcule la división entre las suma de los valores pares y la suma de los valores impares.

Parte 2:

5. Responda las siguientes preguntas:

a. ¿Cuál es la diferencia entre un paso de parámetros por valor y uno por referencia?

b. ¿Cómo se implementa en C un paso de parámetros a una función por referencia?

6. ¿Qué problema tiene la siguiente función? Reescriba el código de la función para arreglarlo, sin cambiar las premisas de uso.

```
char* WordInput(){
   char word[20];
   printf ("Type a word: ");
   scanf ("%s", word);
   return word;
}
```

7. Dadas las siguientes declaraciones en Lenguaje C.

```
int f0[] = { 0, 1, 2, 3, 4},
    f1[] = {10,11,12,13,14},
    f2[] = {20,21,22,23,24},
    f3[] = {30,31,32,33,34},
    f4[] = {40,41,42,43,44},
   *fp[] = {f0,f1,f2,f3,f4},
    **mp = fp;

int ma[][5]={{ 0, 1, 2, 3, 4},{10,11,12,13,14},{20,21,22,23,24},
             {30,31,32,33,34},{40,41,42,43,44}};
```

a. Dibuje las estructuras de datos, partiendo de **mp** y **ma**.

b. escriba funciones en lenguaje C, que imprima por pantalla, a **mp** y **ma**.

8. Dado un string con una secuencia de paréntesis (que abren y cierran en cualquier orden), escriba en C una función que verifiquen que están balanceados.

9. Implemente la función char *trimAll(char *str) en C, que dado un string elimine los blancos al principio, al final y solo deje un blanco entre palabras.

10. Escriba la función HasZeroDiagonal que, dada una matriz de NxN de enteros, retorne 1 si todos los elementos de la diagonal son 0 retorne 0 en caso contrario.