



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



-----**ANÁLISIS DE ALGORITMOS**-----

ACTIVIDAD

Simulación Producto 2 Mayores

PROFESOR:

Franco Martínez Edgardo Adrián

ALUMNO:

Meza Vargas Brandon David – 2020630288

GRUPO:

3CM13



Índice

Problema	3
Mejor Caso	4
Tabla Comparativa	5
Peor Caso	5
Tabla Comparativa	6
Caso Medio	7
Tabla Comparativa	8
Instrucciones para compilar y ejecutar	8

Problema

En el presente ejercicio se pide hacer la simulación del peor, mejor y caso medio del siguiente algoritmo:

```
func Producto2Mayores(A, n)
    if (A[1] > A[2])
        mayor1 = A[1];
        mayor2 = A[2];
    else
        mayor1 = A[2];
        mayor2 = A[1];
    i = 3;

    while (i <= n)
        if (A[i] > mayor1)
            mayor2 = mayor1;
            mayor1 = A[i];
        else if (A[i] > mayor2)
            mayor2 = A[i];
        i = i + 1;

    return mayor1 * mayor2;
```

Imagen 1. Algoritmo producto2Mayores.

Realice una modificación a las comparaciones para considerar a los números que se repiten.

Sabemos que las **operaciones básicas** son: **comparación con los elementos del arreglo** y las **asignaciones a los elementos mayores**.

De igual forma conocemos las funciones para el mejor, peor y caso medio:

- **Peor caso:** $f(n) = 3n - 3$
- **Mejor caso:** $f(n) = 2n - 1$
- **Caso medio:** $f(n) = \frac{8n-7}{3}$

Sabiendo todo lo anterior podemos realizar la simulación realizando los cambios necesarios al código de la imagen 1.

Mejor Caso

El código para este caso se presenta a continuación en la imagen 2.

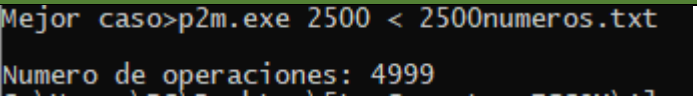
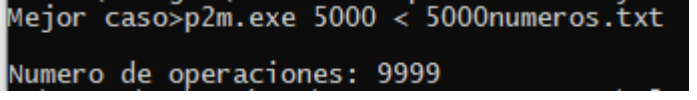
```
7  int main(int argc, char **argv){
8      int n=0, i=0, mayor2, mayor1, res=0, j, aux;
9      int *A; int cont = 0; //variable para contar numero de operaciones
10
11      if(argc != 2 || A == NULL)
12          exit(0);
13
14      n = atoi(argv[1]); //recibimos los n numeros del arreglo por consola
15      A = malloc(n*sizeof(int)); //arreglo dinamico
16
17      //Leemos los n numeros de un archivo
18      for(i=0; i<n; i++){
19          scanf("%d", &A[i]);
20      }
21
22      if(A[0] > A[1]){
23          cont++; //comparacion if
24          mayor1 = A[0]; cont++; //Asignacion
25          mayor2 = A[1]; cont++; //Asignacion
26      }else{
27          cont++;
28          mayor1 = A[1]; cont++; //Asignacion
29          mayor2 = A[0]; cont++; //Asignacion
30      }
31      i = 2;
32
33      while(i < n){
34          cont++; // comparacion if
35          if(A[i] > mayor1){
36              mayor2 = mayor1; cont++; //Asignacion
37              mayor1 = A[i]; cont++; //Asignacion
38          }else{
39              cont++; //comparacion else if
40              if(A[i] > mayor2){
41                  mayor2 = A[i]; cont++; //Asignacion
42              }
43          }
44          i = i + 1;
45      }
46      printf("\nNumero de operaciones: %d", cont);
47      res = mayor1 * mayor2;
48
49      return 0;
50  }
```

Imagen 2. Código mejor caso.

Para que se de el mejor caso, los dos números mayores se deben encontrar en las dos primeras posiciones del arreglo, para esto me auxilie de un archivo que contiene números al azar, de estos números coloque a los dos mayores al inicio. Posteriormente leí el archivo con los números los n números para almacenar en el arreglo.

Para la n 2500, use un archivo que contiene 2500 números, de forma similar, para la n de 5000, use un archivo que contiene 5000 números.

Tabla Comparativa

N	Resultados teóricos	Resultados Prácticos
2500	$f(2500)$ $= 2(2500) - 1$ $= 4999$	
5000	$f(5000)$ $= 2(5000) - 1$ $= 9999$	

Peor Caso

El código para el peor caso se presenta a continuación en la imagen 3.

```

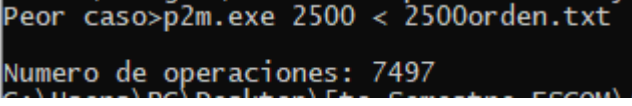
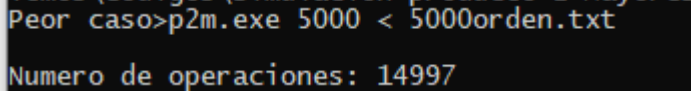
9  int main(int argc, char **argv){
10     int n=0, i=0, mayor2, mayor1, res=0, j, aux;
11     int cont = 0; //variable para contar numero de operaciones
12     int *A;
13
14     if(argc != 2 || A == NULL)
15         exit(0);
16
17     n = atoi(argv[1]); //recibimos los n numeros del arreglo por consola
18     A = malloc(n*sizeof(int)); //arreglo dinamico
19
20     //leemos los n numeros de un archivo
21     for(i=0; i<n; i++){
22         scanf("%d", &A[i]);
23     }
24     if(A[0] > A[1]){
25         cont++; //comparacion if
26         mayor1 = A[0]; cont++; //Asignacion
27         mayor2 = A[1]; cont++; //Asignacion
28     }else{
29         cont++; //comparacion if
30         mayor1 = A[1]; cont++; //Asignacion
31         mayor2 = A[0]; cont++; //Asignacion
32     }
33     i = 2;
34     while(i < n){
35         cont++; // comparacion if
36         if(A[i] > mayor1 || A[i] == mayor1 ){
37             mayor2 = mayor1; cont++; //Asignacion
38             mayor1 = A[i]; cont++; //Asignacion
39         }else{
40             cont++; //comparacion else if
41             if(A[i] > mayor2 || A[i] > mayor2){
42                 mayor2 = A[i]; cont++; //Asignacion
43             }
44         }
45         i = i + 1;
46     }
47     printf("\nNumero de operaciones: %d", cont);

```

Imagen 3. Código peor caso.

Para que se de el peor caso. los dos números mayores deben estar al final del arreglo, para esto se generaron 2500 números aleatorios y se ordenaron de manera ascendente, de igual forma con los 5000 números. Posteriormente leí el archivo con los números los n números para almacenar en el arreglo.

Tabla Comparativa

N	Resultados teóricos	Resultados Prácticos
2500	$f(2500)$ $= 3(2500) - 3$ $= 7497$	 <pre>Peor caso>p2m.exe 2500 < 2500orden.txt Numero de operaciones: 7497</pre>
5000	$f(5000)$ $= 3(5000) - 3$ $= 14997$	 <pre>Peor caso>p2m.exe 5000 < 5000orden.txt Numero de operaciones: 14997</pre>

Caso Medio

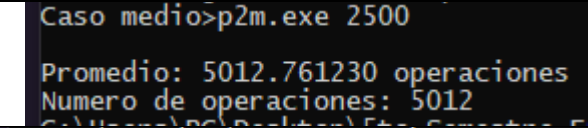
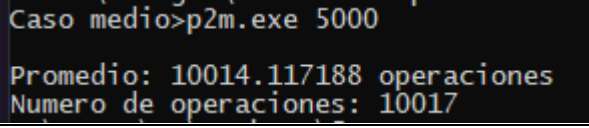
El código para el caso medio se ve en la imagen 4.

```
11  int main(int argc, char **argv){
12      int n=0, i=0, mayor2, mayor1, res=0, j, aux;
13      int cont = 0; //variable para contar numero de operaciones
14      int *A;
15      float promedio = 0;
16
17      if(argc != 2 || A == NULL)
18          exit(0);
19      n = atoi(argv[1]); //recibimos los n numeros del arreglo por consola
20      A = malloc(n*sizeof(int)); //arreglo dinamico
21
22      for(j=0; j<N_VECES; j++){
23          i = 0; cont = 0;
24          //creamos un arreglo con numeros al azar
25          for(i=0; i<n; i++){
26              A[i] = rand() % 30000 + 1;
27          }
28
29          if(A[0] > A[1]){
30              cont++; // comparacion if
31              mayor1 = A[0]; cont++; //Asignacion
32              mayor2 = A[1]; cont++; //Asignacion
33          }else{
34              cont++; // comparacion if
35              mayor1 = A[1]; cont++; //Asignacion
36              mayor2 = A[0]; cont++; //Asignacion
37          }
38          i = 2;
39          while(i < n){
40              cont++; // comparacion if
41              if(A[i] > mayor1 || A[i] == mayor1 ){
42                  mayor2 = mayor1; cont++; //Asignacion
43                  mayor1 = A[i]; cont++; //Asignacion
44              }else{
45                  cont++; //comparacion else if
46                  if(A[i] > mayor2 || A[i] > mayor2){
47                      mayor2 = A[i]; cont++; //Asignacion
48                  }
49              }
50              i = i + 1;
51          }
52          promedio = promedio + (float)cont; //sumamos el promedio
53      }
54      promedio = promedio / N_VECES; //calculamos el promedio
55      printf("\nPromedio: %f operaciones", promedio);
56      printf("\nNumero de operaciones: %d", cont);
57  }
```

Imagen 4. Código del caso medio.

Para obtener el caso medio se hizo el mismo algoritmo del mejor y peor caso, solo que en esta ocasión se hizo n veces, siendo específicos 10, 000 veces como se ve en la imagen 4. De esta forma, al hacerse 10, 000 veces el algoritmo, se pudo obtener el promedio del número de operaciones para el caso medio.

Tabla Comparativa

N	Resultados teóricos	Resultados Prácticos
2500	$f(2500) = \frac{8(2500) - 7}{3}$ $= 6664.33$	 <pre>Caso medio>p2m.exe 2500 Promedio: 5012.761230 operaciones Numero de operaciones: 5012</pre>
5000	$f(5000) = \frac{8(5000) - 7}{3}$ $= 13331$	 <pre>Caso medio>p2m.exe 5000 Promedio: 10014.117188 operaciones Numero de operaciones: 10017</pre>

Instrucciones para compilar y ejecutar

En las carpetas mejor y peor caso encontramos el archivo con extensión .c, además de dos archivos con números, para compilar usamos:

gcc nombre.c -o nombreSalida.exe

Para su ejecución, tenemos que indicar un parámetro que será la n y de que archivo leerá los números:

nombreSalida.exe n < archivo.txt

En la carpeta del caso medio solo encontraremos el archivo .c:

gcc nombre.c -o nombreSalida.exe

Para ejecutarlo solo basta con indicar el parámetro n:

nombreSalida.exe n