

Programacion estructurada.

Daniel Reyes Barrera

2 de diciembre de 2020

Resumen

En este documento se han resuelto algunos problemas computacionales utilizando las herramientas aprendidas en la clase 6 – Punteros, arreglos, uso de memoria. del curso de programación C++, como son direcciones de memoria, punteros, memoria estatica y dinamica, arreglos y arreglos multidimensionales

1. Ejercicio 1.

Hacer un mapa conceptual con los conceptos de esta clase: arreglos unidimensionales, multidimensionales, paso por referencia, paso por valor, memoria estática, memoria dinámica, etc.

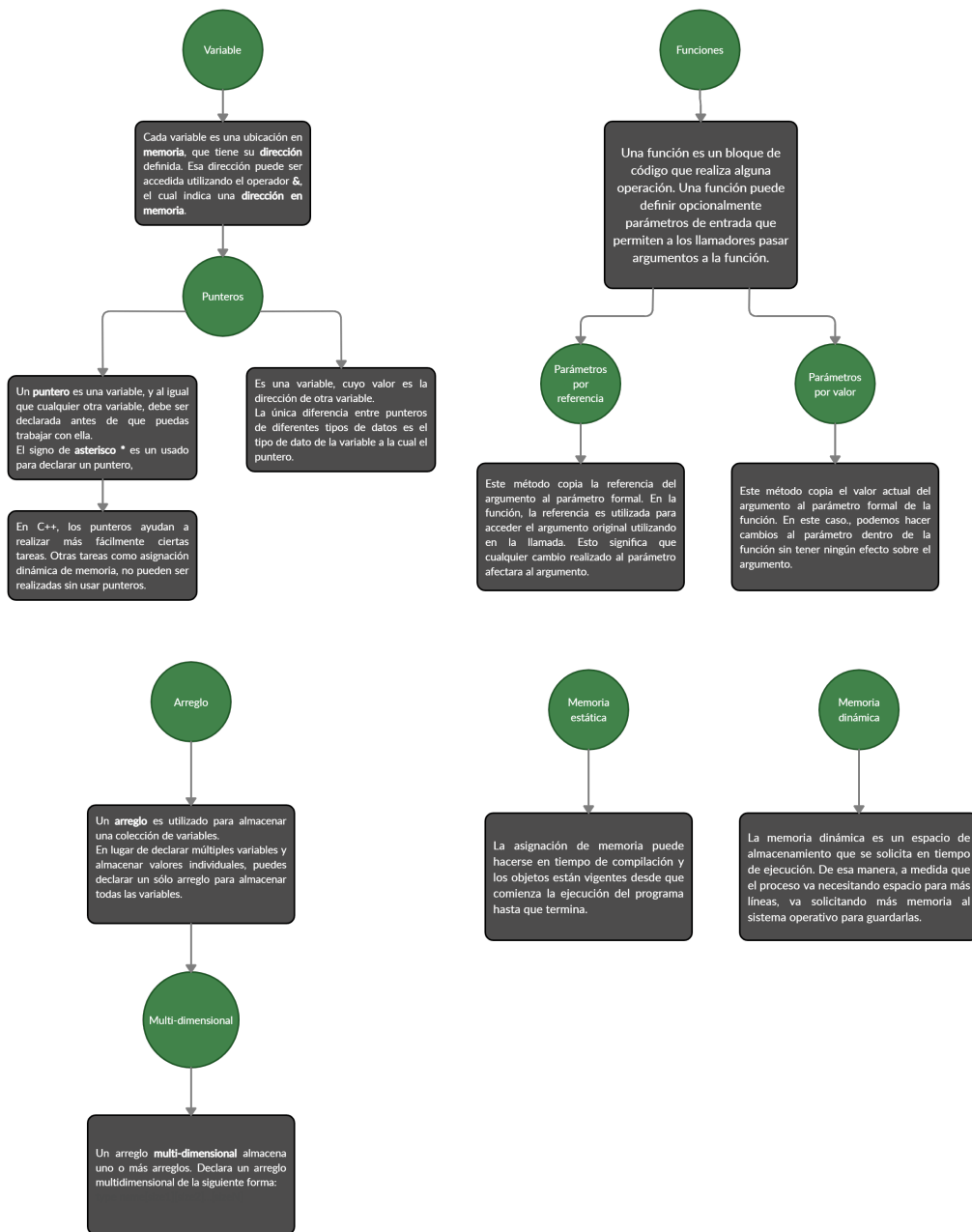


Figura 1: Mapa conceptual

2. Primer programa

El máximo común divisor de dos enteros es el entero más grande que puede dividir a cada uno de los dos números. Programe el algoritmo para calcular el máximo común divisor en un método independiente.

2.1. Problema computacional.

Objetivo: Dado dos número enteros calcular el máximo comun divisor.

Entrada: Dos números enteros.

Salida: El valor de MCD de ambos números introducidos.

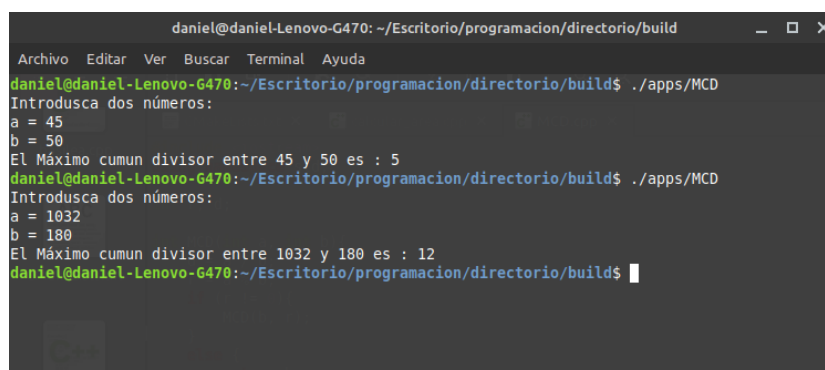
2.2. Algoritmo.

Se reutilizo el código de la tarea 5 y se le hizo una modificación para utilizar punteros, específicamente el paso de parámetros por referencia como apuntadores.

El código fuente del programa se muestra en el Apéndice ??.

2.3. Instancia del problema.

Como prueba de escritorio, se seleccionaron las siguientes instancias del problema. Entrada: $a = 45$, $b = 50$ y $a = 1032$, $b = 180$. La salida del programa se observa en la Figura 2.



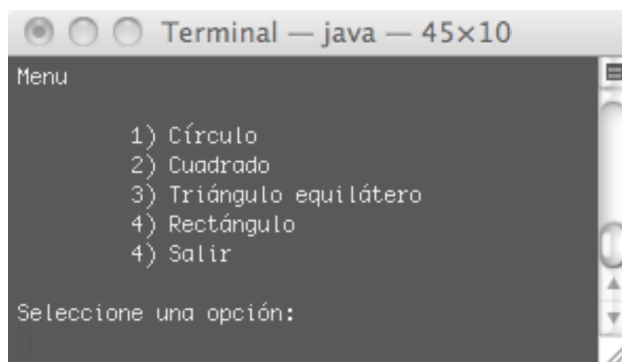
```
daniel@daniel-Lenovo-G470: ~/Escritorio/programacion/directorio/build
Archivo Editar Ver Buscar Terminal Ayuda
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/MCD
Introduzca dos números:
a = 45
b = 50
El Máximo comun divisor entre 45 y 50 es : 5
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/MCD
Introduzca dos números:
a = 1032
b = 180
El Máximo comun divisor entre 1032 y 180 es : 12
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$
```

Figura 2: Ejecución de algunas instancias del problema.

3. Segundo programa

Programa el algoritmo para determinar si el número es palíndromo y el algoritmo para validar la entrada en métodos independientes. Sugerencia: Haga uso de los operadores módulo y división para separar el número tecleado en unidades, decenas, centenas, etc.

Escriba un programa que imprima un menú para seleccionar un tipo de figura geométrica de la siguiente forma:



El usuario debe seleccionar una opción y el programa debe calcular el área y perímetro de la opción seleccionada. Programe cada opción en un método independiente. El programa debe regresar al menu principal hasta que el usuario seleccione la opción salir.

3.1. Problema computacional.

Objetivo: Calcular áreas de figuras deseadas de manera indefinida.

Entrada: Un número que represente la figura deseada y posteriormente sus dimensiones.

Salida: El area de la figura deseada.

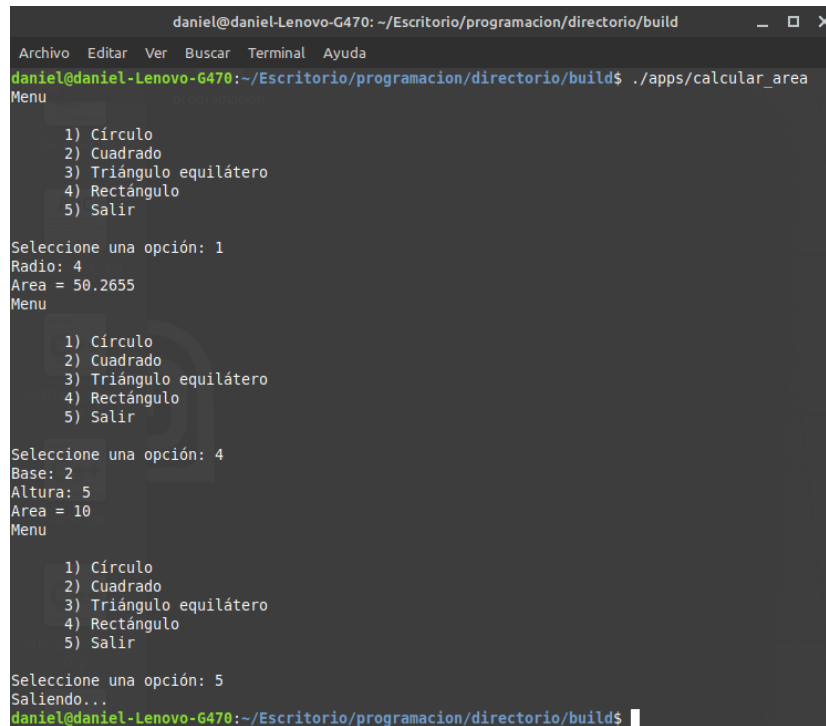
3.2. Algoritmo.

Se reutilizó el código de la tarea 5, reescribiendolo talque se pasará la variable de retorno por referencia.

El código fuente del programa se muestra en el Apéndice 8.

3.3. Instancia del problema.

Como prueba de escritorio, se seleccionaron las siguientes instancias del problema. Se calcularon las siguientes áreas: Un círculo de radio 4 y el área de un rectángulo de base = 2 y altura = 5. La salida del programa se observa en la Figura 3.



```
daniel@daniel-Lenovo-G470: ~/Escritorio/programacion/directorio/build
Archivo Editar Ver Buscar Terminal Ayuda
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/calculador_area
Menu
1) Círculo
2) Cuadrado
3) Triángulo equilátero
4) Rectángulo
5) Salir
Seleccione una opción: 1
Radio: 4
Area = 50.2655
Menu
1) Círculo
2) Cuadrado
3) Triángulo equilátero
4) Rectángulo
5) Salir
Seleccione una opción: 4
Base: 2
Altura: 5
Area = 10
Menu
1) Círculo
2) Cuadrado
3) Triángulo equilátero
4) Rectángulo
5) Salir
Seleccione una opción: 5
Saliendo...
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$
```

Figura 3: Ejecución de algunas instancias del problema.

4. Tercer programa

Programa el algoritmo para determinar si el número es palíndromo y el algoritmo para validar la entrada en métodos independientes. Sugerencia: Haga uso de los operadores módulo y división para separar el número tecleado en unidades, decenas, centenas, etc.

4.1. Problema computacional.

Objetivo: Dado un número entero determinar si es palíndromo.

Entrada: Un número entero mayor que 0.

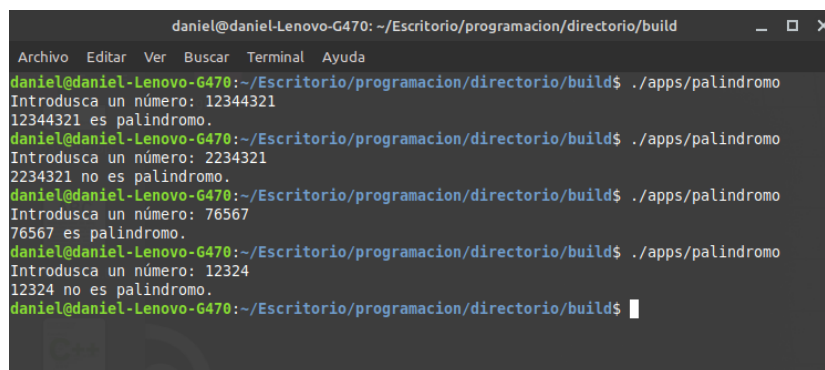
Salida: La respuesta de si el número dado es o no palíndromo.

4.2. Algoritmo.

En este se utilizó el pasar la variable de retorno como apuntador. El código fuente del programa se muestra en el Apéndice 9.

4.3. Instancia del problema.

Como prueba de escritorio, se seleccionaron las siguientes instancias del problema. Entrada: 12344321, 2234321, 76567 y 12324. La salida del programa se observa en la Figura 4.



```
daniel@daniel-Lenovo-G470: ~/Escritorio/programacion/directorio/build
Archivo Editar Ver Buscar Terminal Ayuda
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/palindromo
Introduzca un número: 12344321
12344321 es palindromo.
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/palindromo
Introduzca un número: 2234321
2234321 no es palindromo.
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/palindromo
Introduzca un número: 76567
76567 es palindromo.
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/palindromo
Introduzca un número: 12324
12324 no es palindromo.
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$
```

Figura 4: Ejecución de algunas instancias del problema.

5. Cuarto programa

Escriba un programa que capture 15 números y los imprima ordenados de menor a mayor.

5.1. Problema computacional.

Objetivo: Ordenar una cantidad de n números dados por el usuario.

Entrada: Una serie de números.

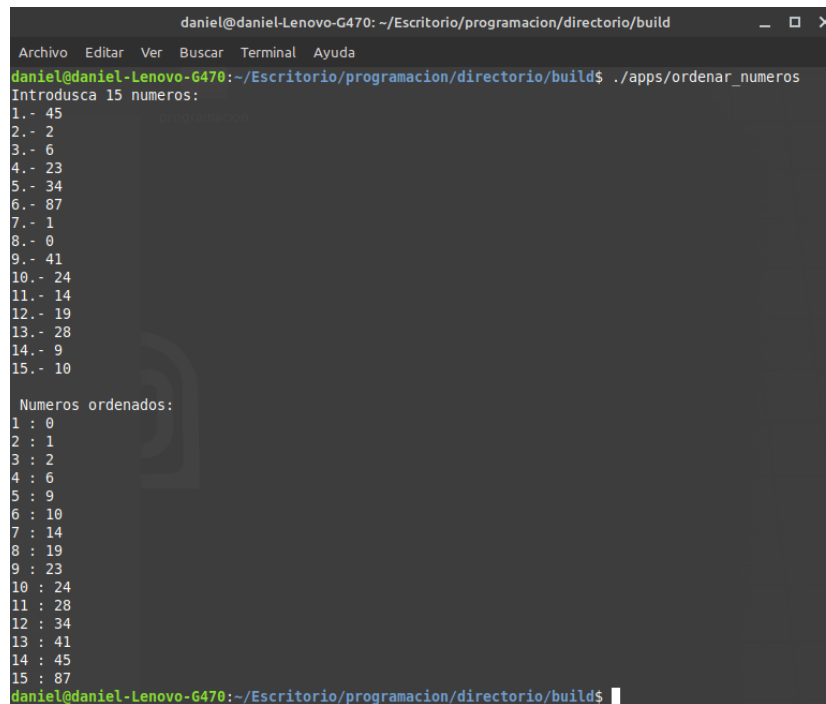
Salida: Los números dados por el usuario impresos de menor a mayor.

5.2. Algoritmo.

Se utilizó un arreglo dinámico para el ordenamiento de los números. El código fuente del programa se muestra en el Apéndice 10.

5.3. Instancia del problema.

Como prueba de escritorio, se seleccionó la siguiente instancia del problema. Entrada: 45, 2, 6, 23, 34, 87, 1, 0, 41, 24, 14, 19, 28, 9 y 10. La salida del programa se observa en la Figura 5.



```
daniel@daniel-Lenovo-G470: ~/Escritorio/programacion/directorio/build
Archivo Editar Ver Buscar Terminal Ayuda
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$ ./apps/ordenar_numeros
Introduzca 15 numeros:
1.- 45
2.- 2
3.- 6
4.- 23
5.- 34
6.- 87
7.- 1
8.- 0
9.- 41
10.- 24
11.- 14
12.- 19
13.- 28
14.- 9
15.- 10

Numeros ordenados:
1 : 0
2 : 1
3 : 2
4 : 6
5 : 9
6 : 10
7 : 14
8 : 19
9 : 23
10 : 24
11 : 28
12 : 34
13 : 41
14 : 45
15 : 87
daniel@daniel-Lenovo-G470:~/Escritorio/programacion/directorio/build$
```

Figura 5: Ejecución de algunas instancias del problema.

6. Conclusiones.

El programar con apuntadores y tener en cuenta la memoria estática y dinámica tiene muchos beneficios cuando se programa un gran algoritmo. Otra de las grandes ventajas de la utilización de punteros es la posibilidad de realizar una asignación dinámica de memoria. Esto significa que la reserva de memoria se realiza dinámicamente en tiempo de ejecución, no siendo necesario entonces tener que especificar en la declaración de variables la cantidad de memoria que se va a requerir. La reserva de memoria dinámica añade una gran flexibilidad a los programas porque permite al programador la posibilidad de reservar la cantidad de memoria exacta en el preciso instante en el que se necesite, sin tener que realizar una reserva por exceso en prevención a la que pueda llegar a necesitar.

7. Código fuente de MCD


```

1  #include <iostream>
2  using namespace std;
3
4  int r, d;
5
6  void MCD(int a, int b, int *res)
7  {
8      d = a / b;
9      r = a % b;
10     if (r != 0)
11     {
12         MCD(b, r, res);
13     }
14     else
15     {
16         *res = b;
17     }
18 }
19
20 int main()
21 {
22     int a, b, res;
23
24     cout << "Introduzca dos numeros: \na=";
25     cin >> a;
26     cout << "b=";
27     cin >> b;
28     MCD(a, b, &res);
29     cout << "El Maximo comun divisor entre "
30     << a << " y " << b << " es: " << res << endl;
31
32     return 0;
33 }

```

8. Código fuente de calcular areas

```

1  #include <iostream>
2  #include <cmath>
3
4  using namespace std;

```

```

5
6 float &circulo(float &x){
7     x = M_PI * x * x;
8     return x;
9 }
10
11 float &cuadrado(float &x){
12     x = x * x;
13     return x;
14 }
15
16 float &triangulo(float &x){
17     x = x * x * sqrt(3) / 4;
18     return x;
19 }
20
21 float &rectangulo(float &x, float &y){
22     x = x * y;
23     return x;
24 }
25
26 int main(){
27     int objeto;
28     float x, y;
29     bool no_salir = true;
30     do
31     {
32
33         cout << "Menu\n\n.....1) Circulo\n....."
34         << "2) Cuadrado\n.....3) Triingulo equilatero\n"
35         << ".....4) Rectangulo\n.....5) Salir\n"
36         << "\nSeleccione una opcion:\n";
37         cin >> objeto;
38         switch (objeto) {
39             case 1:
40                 cout << "Radio: ";
41                 cin >> x;
42                 cout << "Area= " << circulo(x) << endl;
43                 break;
44

```

```

45         case 2:
46             cout << "Lado: ";
47             cin >> x;
48             cout << "Area=" << cuadrado(x) << endl;
49             break;
50
51         case 3:
52             cout << "Lado: ";
53             cin >> x;
54             cout << "Area=" << triangulo(x) << endl;
55
56             break;
57
58         case 4:
59             cout << "Base: ";
60             cin >> x;
61             cout << "Altura: ";
62             cin >> y;
63             cout << "Area=" << rectangulo(x, y) << endl;
64             break;
65
66         case 5:
67             no_salir = false;
68             cout << "Saliendo ... \n";
69             break;
70
71         default:
72             cout << "Entrada invalida \n";
73             no_salir = false;
74             break;
75     }
76
77     } while (no_salir);
78
79     return 0;
80 }

```

9. Código fuente de Palindromo

```

1 #include <iostream>

```

```

2  using namespace std;
3
4  bool *palindromo(int n, bool *x) {
5
6      int a = n, b = 0;
7      while (a > 0) {
8
9          b *= 10;
10         b += a % 10;
11         a = (a - a % 10) / 10;
12     }
13     if (b == n) {
14         *x = true;
15         return x;
16     }
17     else {
18         *x = false;
19         return x;
20     }
21 }
22
23
24 int main(){
25
26     int n;
27
28     cout << "Introduzca un numero: ";
29     cin >> n;
30     bool *x = new bool;
31     if (*palindromo(n, x)) {
32         cout << n << " es palindromo." << endl;
33     }
34     else {
35         cout << n << " no es palindromo." << endl;
36     }
37     delete x;
38
39     return 0;
40 }

```

10. Código fuente del Ordenar números

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6
7     int *array = new int [15];
8     cout << "Introduzca 15 números: \n";
9     for (int i = 0; i < 15; i++)
10    {
11        cout << i + 1 << ".- ";
12        cin >> array[i];
13    }
14    for (int k = 0; k < 15; k++)
15    {
16        int menor = k;
17
18        for (int j = k + 1; j < 15; j++)
19        {
20            if (*(array + menor) > *(array + j))
21            {
22                menor = j;
23            }
24        }
25        int g = array[k];
26        *(array + k) = *(array + menor);
27        *(array + menor) = g;
28    }
29    cout << "\n Números ordenados: \n";
30    for (int jj = 0; jj < 15; jj++)
31    {
32        cout << jj + 1 << " : " << *(array + jj) << endl;
33    }
34    delete array;
35    return 0;
36 }
```