**MAKALAH PRAKTIKUM**

**LAB DESAIN DAN ANALISIS ALGORTIMA**

****

Disusun oleh :

**Muhammad Toriq Robbani**

**TI 23 PA2**

**232310035**

**PROGRAM STUDI TEKNOLOGI INFORMASI**

**FAKULTAS INFORMATIKA DAN PARIWISATA  
INSTITUT BISNIS DAN INFORMATIKA KESATUAN BOGOR**

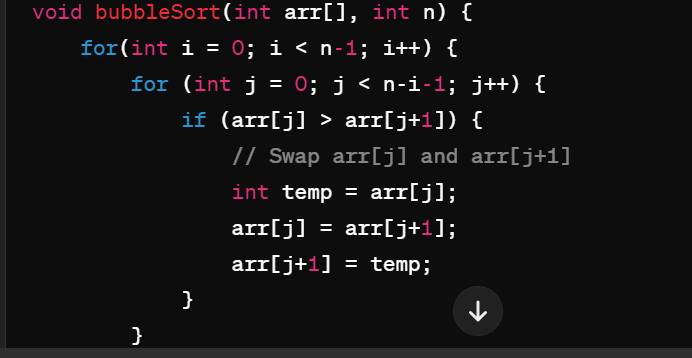
**2024**

# Pengurutan (Sorting)

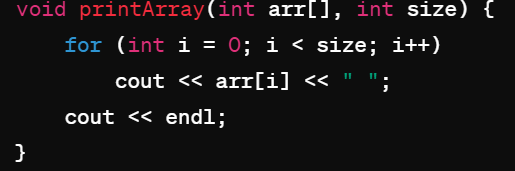
## Bubble Sort

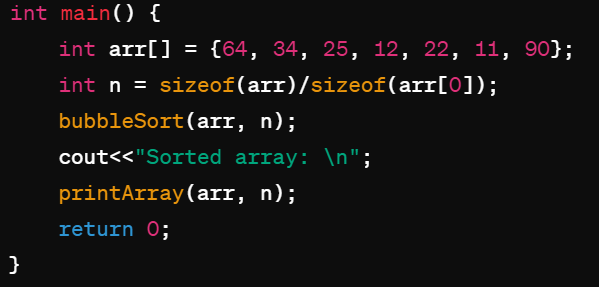
## 

* + - **include<iostream>:** Ini adalah direktif preprosesor yang memasukkan header file iostream, yang memungkinkan penggunaan fungsi input-output standar C++ seperti cout dan endl.
    - **using namespace std;**: Ini mengindikasikan bahwa kita akan menggunakan namespace std, yang berarti kita tidak perlu menuliskan std:: sebelum menggunakan fungsi-fungsi dari namespace tersebut.



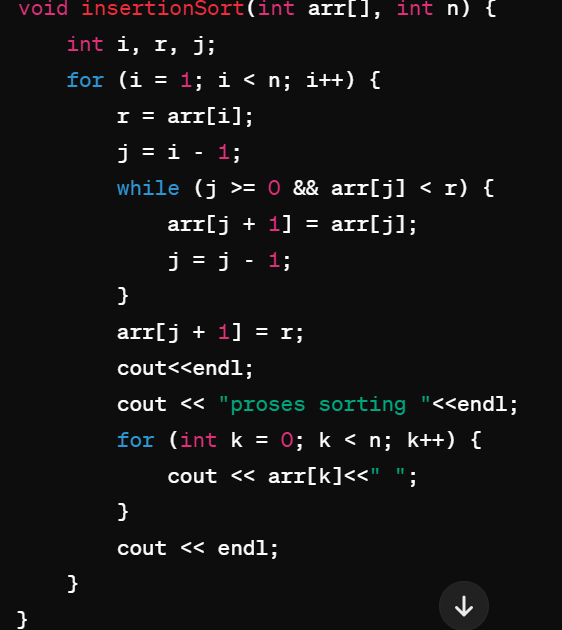
* + - **bubbleSort**: Ini adalah fungsi yang mengimplementasikan algoritma Bubble Sort. Fungsi ini menerima dua parameter: sebuah array arr yang akan diurutkan dan panjang array n.
    - Algoritma Bubble Sort bekerja dengan membandingkan dua elemen berturut-turut dan menukar mereka jika urutannya tidak benar. Ini dilakukan secara berulang sampai seluruh array diurutkan.
    - Dua loop bersarang digunakan di sini untuk melakukan iterasi melalui array. **Loop pertama (i)** digunakan untuk mengontrol jumlah langkah iterasi dan **loop kedua (j)** digunakan untuk membandingkan dan menukar elemen.



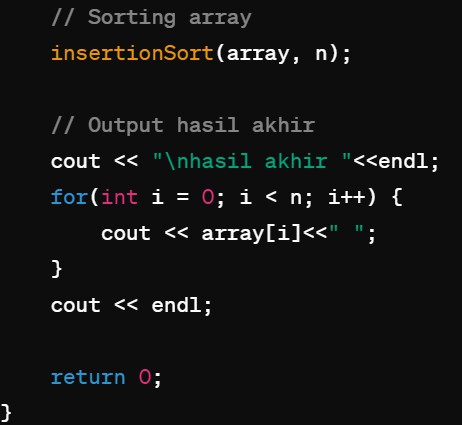
* + - **printArray**: Ini adalah fungsi yang digunakan untuk mencetak isi dari array yang diurutkan.
    - Fungsi ini menerima dua parameter: sebuah array arr yang akan dicetak dan ukuran array size.
    - Fungsi ini mengiterasi melalui array dan mencetak setiap elemen dengan spasi sebagai pemisah, kemudian mencetak **newline (endl)** setelah semua elemen dicetak.
    - **main**: Ini adalah fungsi utama yang akan dieksekusi pertama kali saat program dijalankan.
    - Di dalam fungsi main, sebuah array integer arr yang belum diurutkan diinisialisasi dengan beberapa nilai.
    - Kemudian, panjang array (n) dihitung dengan membagi ukuran total array dengan ukuran elemen tunggalnya.
    - Fungsi bubbleSort dipanggil untuk mengurutkan array arr.
    - Setelah array diurutkan*, pesan* **"Sorted array:"** dicetak, diikuti dengan panggilan ke fungsi printArray untuk mencetak array yang sudah diurutkan.
    - Program kemudian mengembalikan nilai 0 untuk menandakan bahwa program telah berakhir dengan sukses.

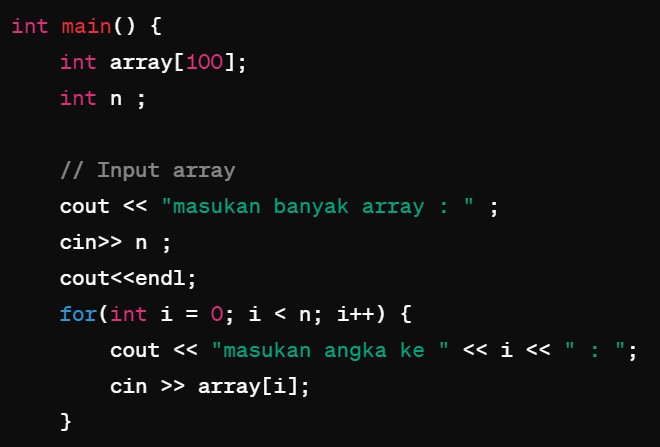
## Insertion Sort

* + - **#include <iostream>**: Ini adalah direktif preprosesor yang memasukkan header file iostream, yang memungkinkan penggunaan fungsi input-output standar C++ seperti cout dan cin.
    - **using namespace std;**: Ini mengindikasikan bahwa kita akan menggunakan namespace std, yang berarti kita tidak perlu menuliskan std:: sebelum menggunakan fungsi-fungsi dari namespace tersebut.



* + - **insertionSort**: Ini adalah fungsi yang mengimplementasikan algoritma Insertion Sort. Fungsi ini menerima dua parameter: sebuah array arr yang akan diurutkan dan panjang array n.
    - Algoritma Insertion Sort bekerja dengan memilih satu elemen pada suatu waktu dan memasukkannya ke dalam posisi yang tepat dalam urutan yang sudah diurutkan.
    - Loop eksternal digunakan untuk mengatur elemen mana yang akan dimasukkan pada setiap iterasi.
    - Pada setiap iterasi, elemen yang akan dimasukkan disimpan dalam variabel r, dan loop internal digunakan untuk memindahkan elemen- elemen yang lebih besar dari r ke kanan.
    - Setelah menemukan posisi yang tepat, elemen r dimasukkan ke posisi yang sesuai dalam urutan.
    - Selama proses sorting, setiap langkah iterasi dicetak untuk menampilkan bagaimana array berubah pada setiap iterasi.

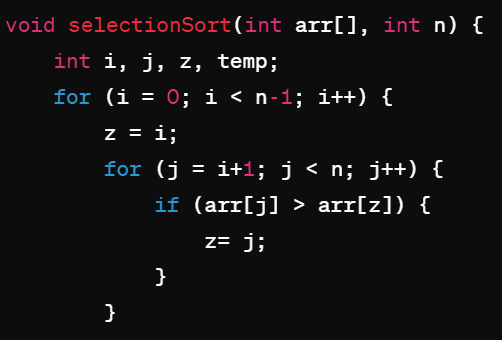
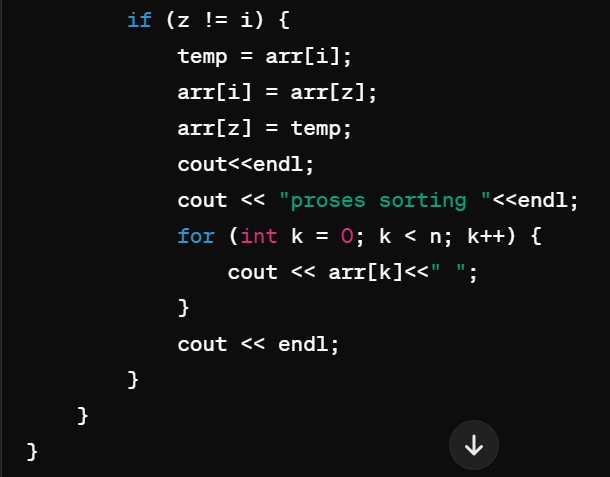




* + - **main**: Ini adalah fungsi utama yang akan dieksekusi pertama kali saat program dijalankan.
    - Di dalam fungsi main, array array dengan kapasitas maksimum 100 dan sebuah variabel n untuk menyimpan jumlah elemen array dimuat.
    - Pengguna diminta untuk memasukkan jumlah elemen array (n) dan nilai- nilai elemen array.
    - Setelah array dimuat, fungsi insertionSort dipanggil untuk mengurutkan array tersebut.
    - Setelah pengurutan selesai, hasil akhir dari array dioutputkan ke layar.

## Selection Sort

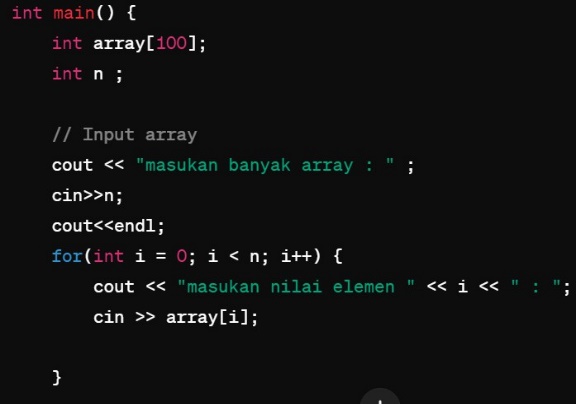
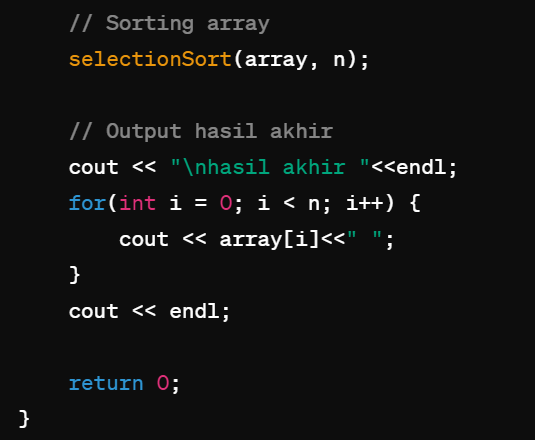
* + - **#include <iostream>**: Ini adalah direktif preprosesor yang memasukkan header file iostream, yang memungkinkan penggunaan fungsi input-output standar C++ seperti cout dan cin.
    - **using namespace std;**: Ini mengindikasikan bahwa kita akan menggunakan namespace std, yang berarti kita tidak perlu menuliskan std:: sebelum menggunakan fungsi-fungsi dari namespace tersebut.



* + - **selectionSort**: Ini adalah fungsi yang mengimplementasikan algoritma Selection Sort. Fungsi ini menerima dua parameter: sebuah array arr yang akan diurutkan dan panjang array n.
    - Algoritma Selection Sort bekerja dengan mencari elemen terbesar (atau terkecil) dalam array dan memindahkannya ke posisi yang sesuai dalam urutan yang sudah diurutkan.
    - Dua loop bersarang digunakan di sini. **Loop pertama (i)** digunakan untuk mengatur posisi elemen yang akan dipertukarkan, sedangkan **loop kedua**

**(j)** digunakan untuk mencari elemen terbesar yang belum diurutkan.

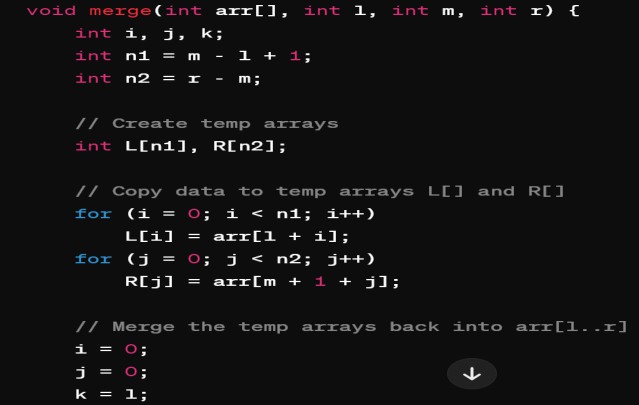
* + - Setelah menemukan elemen terbesar yang belum diurutkan, elemen tersebut ditukar dengan elemen pertama dalam urutan yang belum diurutkan.
    - Setelah proses pertukaran, proses sorting (urutan array saat itu) dicetak ke layar.

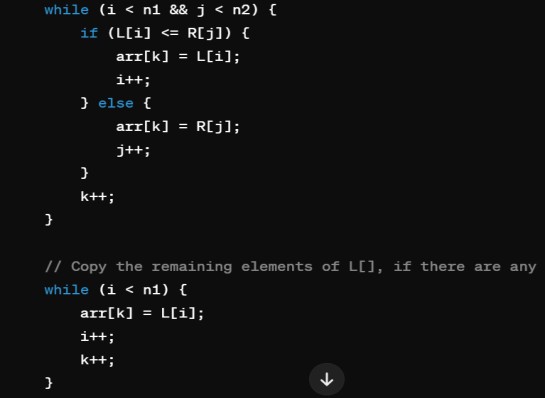
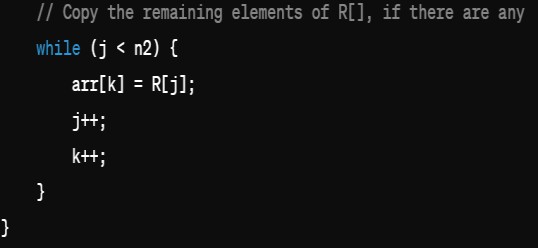
 

* + - **main**: Ini adalah fungsi utama yang akan dieksekusi pertama kali saat program dijalankan.
    - Di dalam fungsi main, array array dengan kapasitas maksimum 100 dan sebuah variabel n untuk menyimpan jumlah elemen array dimuat.
    - Pengguna diminta untuk memasukkan jumlah elemen array (n) dan nilai- nilai elemen array.
    - Setelah array dimuat, fungsi selectionSort dipanggil untuk mengurutkan array tersebut.
    - Setelah pengurutan selesai, hasil akhir dari array dioutputkan ke layar.

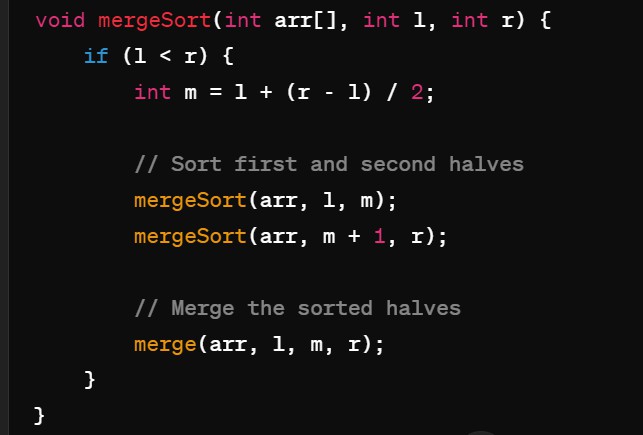
# Pengurutan Lanjutan

## Merge Sort

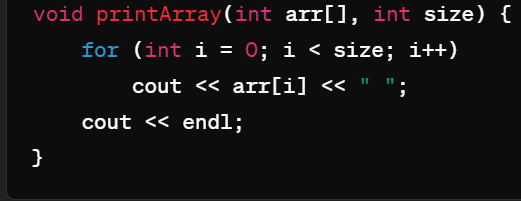
* + - **#include <iostream>**: Ini adalah direktif preprosesor yang memasukkan header file iostream, yang memungkinkan penggunaan fungsi input-output standar C++ seperti cout.
    - **using namespace std;**: Ini mengindikasikan bahwa kita akan menggunakan namespace std, yang berarti kita tidak perlu menuliskan std:: sebelum menggunakan fungsi-fungsi dari namespace tersebut.



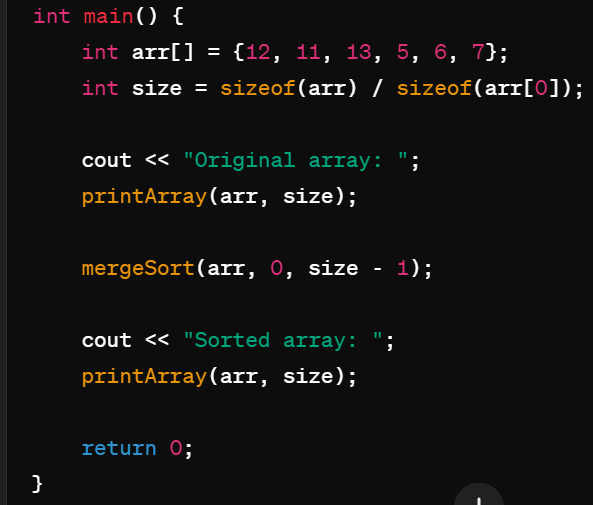
* + - **merge**: Ini adalah fungsi yang digunakan untuk menggabungkan dua sub- array yang sudah diurutkan menjadi satu array hasil pengurutan.
    - Fungsi ini menerima empat parameter: array arr, indeks awal l, indeks tengah m, dan indeks akhir r.
    - Pertama, fungsi ini menghitung ukuran masing-masing sub-array (n1 dan n2).
    - Kemudian, dua sub-array sementara **(L dan R)** dibuat untuk menyimpan elemen-elemen dari sub-array yang sudah diurutkan.
    - Selanjutnya, elemen-elemen dari sub-array yang sudah diurutkan disalin ke sub-array sementara L dan R.
    - Setelah itu, elemen-elemen dari kedua sub-array sementara digabungkan kembali ke dalam array utama arr secara terurut.



* + - **mergeSort**: Ini adalah fungsi yang mengimplementasikan algoritma Merge Sort.
    - Fungsi ini menerima tiga parameter: array arr, indeks awal l, dan indeks akhir r.
    - Fungsi ini menggunakan pendekatan rekursif untuk mengurutkan array.
    - Pertama, fungsi ini membagi array menjadi dua bagian seimbang menggunakan **indeks tengah m**.
    - Kemudian, masing-masing bagian array diurutkan secara terpisah dengan memanggil fungsi mergeSort secara rekursif.
    - Setelah itu, kedua bagian yang sudah diurutkan digabungkan kembali menggunakan fungsi merge.

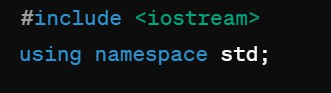


* + - **printArray**: Ini adalah fungsi yang digunakan untuk mencetak isi dari array.
    - Fungsi ini menerima dua parameter: array arr yang akan dicetak dan ukuran array size.
    - Fungsi ini mengiterasi melalui array dan mencetak setiap elemen dengan spasi sebagai pemisah, kemudian mencetak newline setelah semua elemen dicetak.

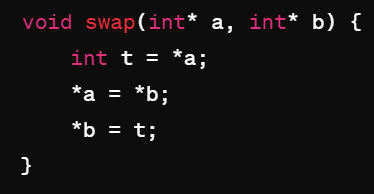


* + - **main**: Ini adalah fungsi utama yang akan dieksekusi pertama kali saat program dijalankan.
    - Di dalam fungsi main, sebuah array integer arr yang belum diurutkan diinisialisasi dengan beberapa nilai.
    - Ukuran array **(size)** dihitung dengan membagi ukuran total array dengan ukuran elemen tunggalnya.
    - Pesan "Original array:" dicetak diikuti dengan panggilan ke fungsi printArray untuk mencetak array yang belum diurutkan.
    - Fungsi mergeSort dipanggil untuk mengurutkan array arr.
    - Setelah array diurutkan, pesan **"Sorted array:"** dicetak diikuti dengan panggilan ke fungsi printArray untuk mencetak array yang sudah diurutkan.
    - Program kemudian mengembalikan nilai 0 untuk menandakan bahwa program telah berakhir dengan sukses.

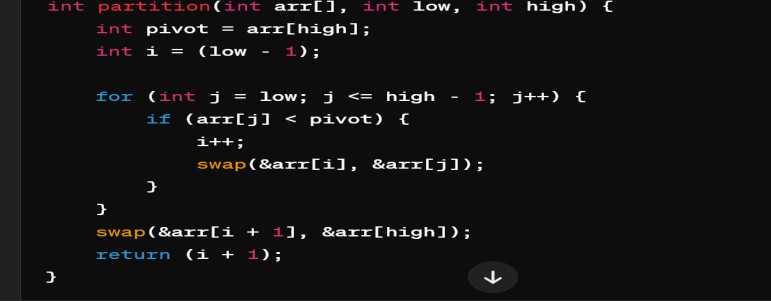
## Quick Sort



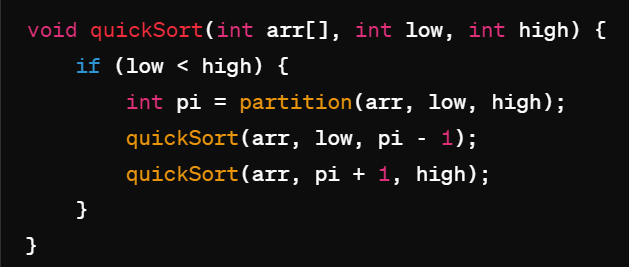
* + - **#include <iostream>**: Ini adalah direktif preprosesor yang memasukkan header file iostream, yang memungkinkan penggunaan fungsi input-output standar C++ seperti cout.
    - **using namespace std;**: Ini mengindikasikan bahwa kita akan menggunakan namespace std, yang berarti kita tidak perlu menuliskan std:: sebelum menggunakan fungsi-fungsi dari namespace tersebut.



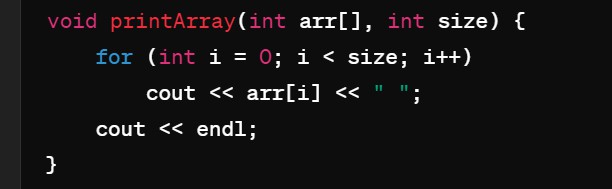
* + - **swap**: Ini adalah fungsi yang digunakan untuk menukar nilai dua variabel yang diberikan.
    - **Fungsi ini menerima dua parameter**: pointer ke dua variabel a dan b.
    - Nilai dari variabel a disimpan dalam variabel sementara t, kemudian nilai a diperbarui dengan nilai dari b, dan nilai b diperbarui dengan nilai dari t, sehingga nilai a dan b ditukar.



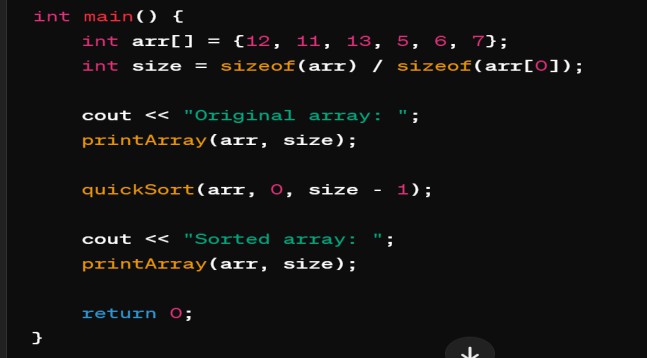
* + - **partition**: Ini adalah fungsi yang digunakan dalam algoritma Quick Sort untuk membagi array menjadi dua sub-array berdasarkan pivot.
    - Fungsi ini menerima tiga parameter: array arr, indeks awal low, dan indeks akhir high dari sub-array yang akan dipartisi.
    - Pada awalnya, pivot dipilih dari elemen terakhir dari array.
    - Selama iterasi melalui array dari low hingga high-1, elemen-elemen yang lebih kecil dari pivot dipindahkan ke bagian kiri dari pivot.
    - Setelah iterasi selesai, elemen pivot dipindahkan ke posisi yang tepat dalam array.
    - Indeks dari pivot setelah dipartisi dikembalikan.



* + - **quickSort**: Ini adalah fungsi utama yang mengimplementasikan algoritma Quick Sort.
    - Fungsi ini menggunakan pendekatan rekursif untuk mengurutkan array.
    - Pada setiap panggilan rekursif, array dipartisi di sekitar sebuah pivot menggunakan fungsi partition.
    - Setelah dipartisi, dua bagian array dipilah untuk diurutkan secara rekursif.

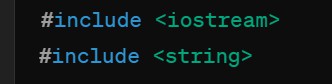


* + - **printArray**: Ini adalah fungsi yang digunakan untuk mencetak isi dari array.
    - Fungsi ini menerima dua parameter: array arr yang akan dicetak dan ukuran array size.
    - Fungsi ini mengiterasi melalui array dan mencetak setiap elemen dengan spasi sebagai pemisah, kemudian mencetak newline setelah semua elemen dicetak.

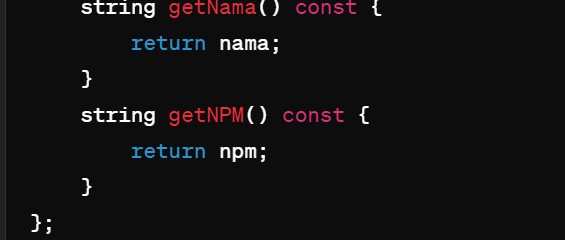
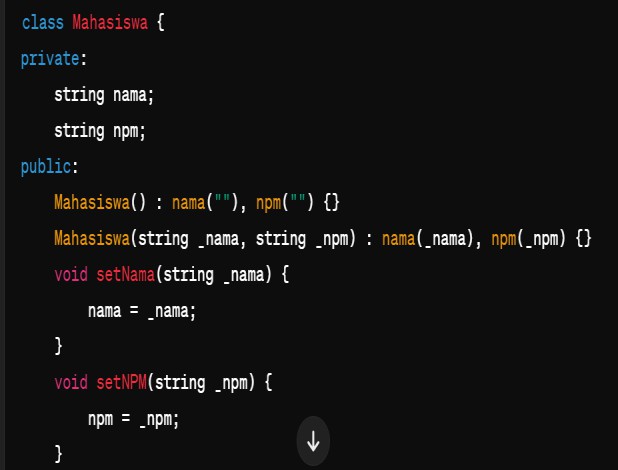


* + - Pada fungsi main, sebuah array arr dengan elemen yang belum terurut dideklarasikan.
    - Ukuran array dihitung menggunakan ekspresi **sizeof(arr) / sizeof(arr[0]).**
    - Array awal dicetak.
    - Fungsi quickSort dipanggil untuk **mengurutkan array**.
    - Array yang telah terurut dicetak kembali.

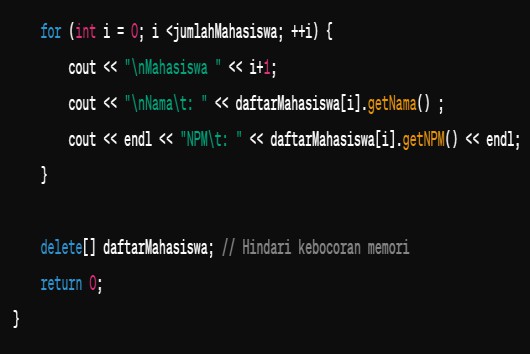
# Class (Setter & Getter)

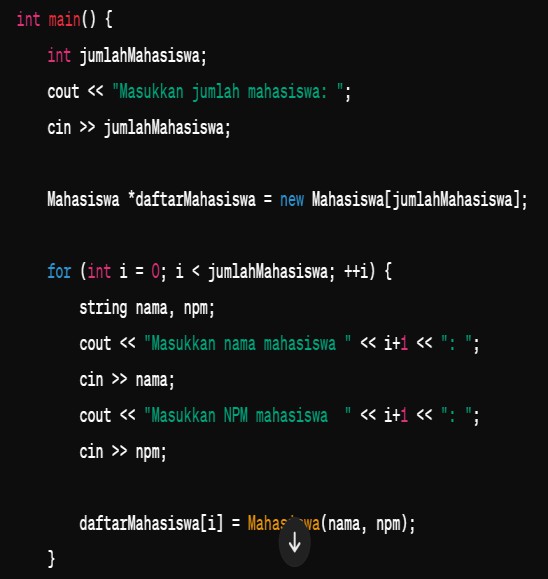


* Mendefinisikan dua header file, yaitu **<iostream>** untuk operasi input/output standar dan
* **<string>** untuk penggunaan tipe data string.
* Menggunakan **namespace std** untuk menghindari penulisan std:: sebelum fungsi dari library standar.



* Kelas Mahasiswa memiliki dua atribut nama dan npm yang menyimpan nama dan NPM mahasiswa.
* Terdapat dua konstruktor: satu tanpa parameter untuk inisialisasi atribut dengan nilai default, dan satu dengan parameter untuk inisialisasi atribut dengan nilai yang diberikan.
* Metode **setNama dan setNPM** digunakan untuk mengatur nilai atribut nama dan npm.
* Metode **getNama dan getNPM** digunakan untuk mengembalikan nilai atribut nama dan npm.

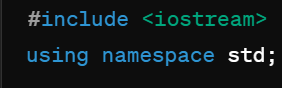




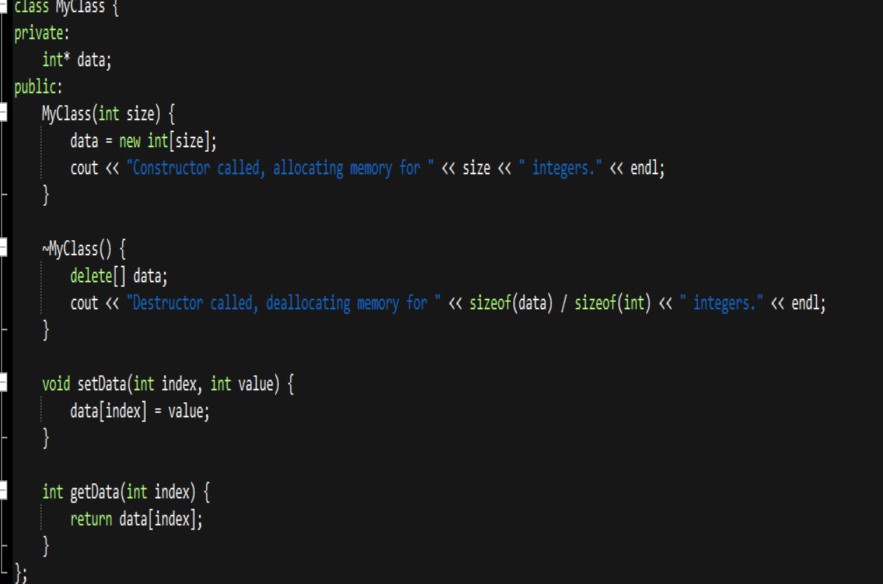
* Di dalam fungsi main, pengguna diminta untuk memasukkan jumlah mahasiswa.
* Selanjutnya, dilakukan alokasi memori dinamis untuk array daftarMahasiswa yang memiliki ukuran sesuai dengan jumlah mahasiswa yang dimasukkan.
* Melalui **loop for**, pengguna diminta untuk memasukkan nama dan NPM masing-masing mahasiswa, dan objek Mahasiswa yang sesuai dibuat dan disimpan di dalam array.
* Setelah semua data dimasukkan, dilakukan loop kembali untuk mencetak semua data mahasiswa yang telah dimasukkan.
* Terakhir, array dinamis daftarMahasiswa dihapus untuk mencegah kebocoran memori.

# Class (Destructor & Constructor)

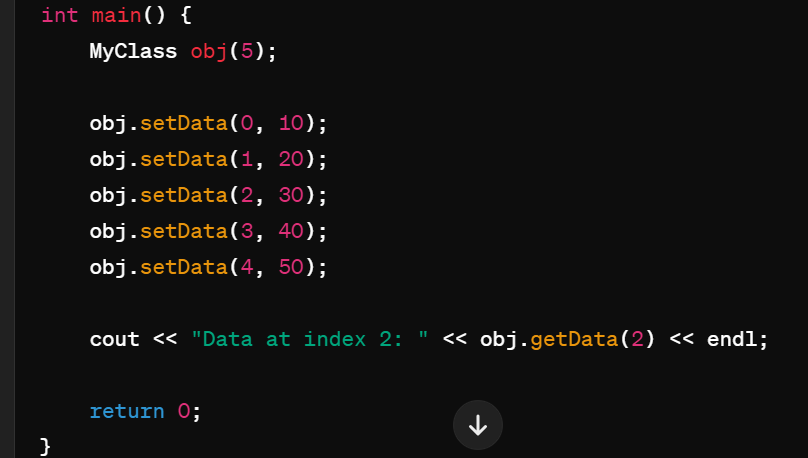
## Destructor



* Ini adalah bagian standar yang mengimpor pustaka input/output stream **(iostream)**
* menggunakan **namespace std** untuk mempermudah penggunaan objek dan fungsi standar dari C++.

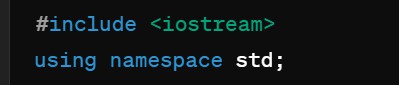


* **MyClass** adalah nama kelas yang memiliki satu atribut pribadi (private), yaitu pointer ke int yang disebut data.
* Konstruktor MyClass dialokasikan memori dinamis untuk array int dengan ukuran size, kemudian mencetak pesan bahwa konstruktor dipanggil.
* Destruktor ~MyClass menghapus memori yang dialokasikan untuk data saat objek dihapus, kemudian mencetak pesan bahwa destruktor dipanggil.
* setData digunakan untuk mengatur nilai elemen array di indeks tertentu.
* getData digunakan untuk mendapatkan nilai elemen array di indeks tertentu.

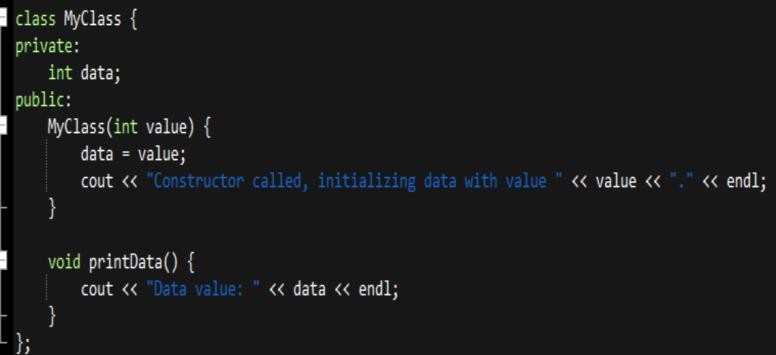


* **Objek obj dari kelas MyClass** dibuat dengan alokasi memori untuk 5 integer.
* Nilai pada indeks tertentu diatur menggunakan setData.
* Nilai pada indeks 2 diambil dan dicetak menggunakan getData.
* Setelah main selesai, destruktor untuk objek obj dipanggil otomatis, yang akan menghapus memori yang dialokasikan untuk array data.

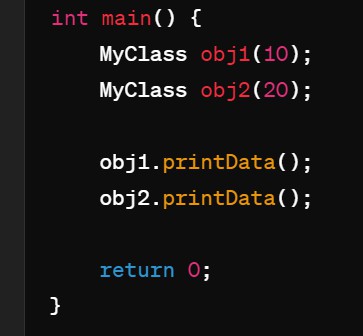
## Constructor



* Ini adalah bagian standar yang mengimpor pustaka input/output stream **(iostream)**
* menggunakan namespace std untuk mempermudah penggunaan objek dan fungsi standar dari C++.

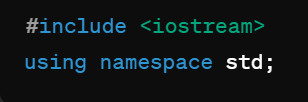


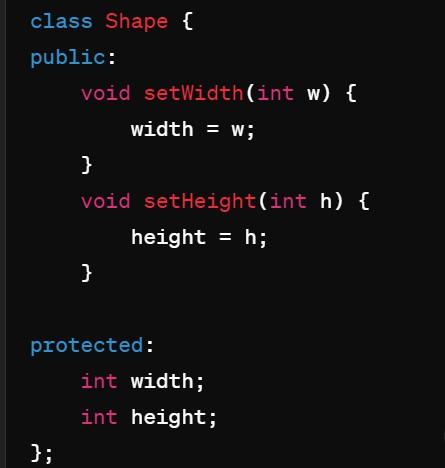
* **MyClass** adalah nama kelas yang memiliki satu atribut pribadi (private), yaitu data bertipe integer.
* Konstruktor MyClass menerima satu parameter value dan menginisialisasi data dengan nilai tersebut, kemudian mencetak pesan bahwa konstruktor dipanggil.
* Metode **printData** digunakan untuk mencetak nilai data.



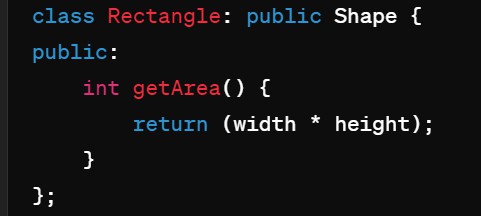
* Dua objek **obj1 dan obj2** dari kelas MyClass dibuat dengan menggunakan konstruktor. Saat objek dibuat, konstruktor dipanggil untuk menginisialisasi atribut data masing-masing objek dengan nilai yang diberikan.
* Kemudian, metode printData dipanggil untuk masing-masing objek, yang mencetak nilai data yang telah diinisialisasi sebelumnya.

# Class (Konsep OOP)

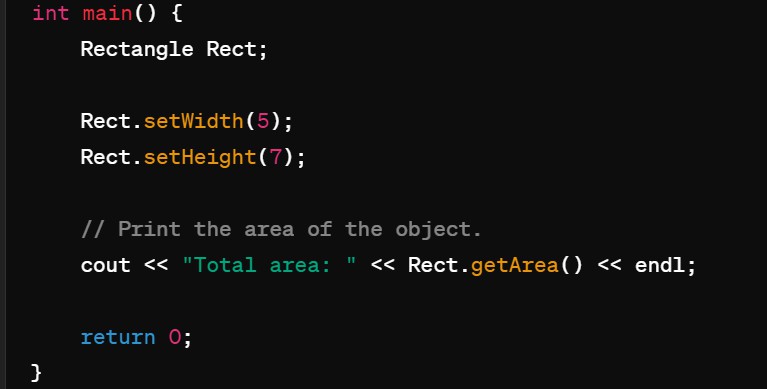
* 1. **Inheritance**
* Baris pertama **(#include <iostream>)** adalah preprocessor directive untuk menyertakan library iostream, yang digunakan untuk operasi input-output dalam C++.
* **using namespace std;** memungkinkan penggunaan istilah-istilah standar dari namespace std tanpa menyertakan std:: sebelumnya.



* **Kelas Shape** adalah kelas dasar yang memiliki dua variabel anggota width dan height, serta metode untuk mengatur **(setWidth() dan setHeight())** nilai- nilai variabel tersebut.
* Variabel anggota **(width dan height)** dinyatakan sebagai protected, sehingga dapat diakses oleh kelas turunan.



* Kelas Rectangle memiliki metode getArea() yang menghitung dan mengembalikan luas persegi panjang, yaitu produk dari width dan height dari kelas Shape.
* Kelas Rectangle merupakan kelas turunan dari kelas Shape, yang dinyatakan dengan kata kunci public setelah titik dua (:).
* Ini berarti kelas Rectangle mewarisi semua anggota dan metode publik dari kelas Shape.

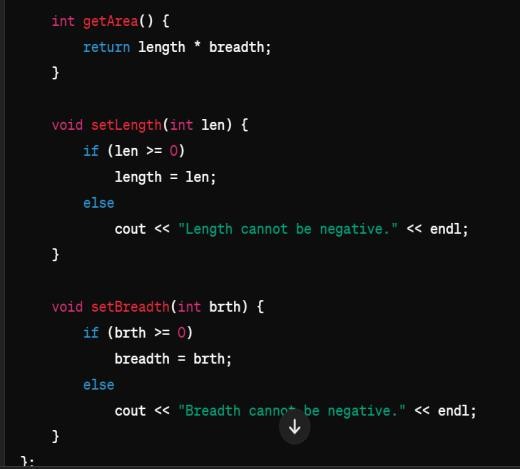
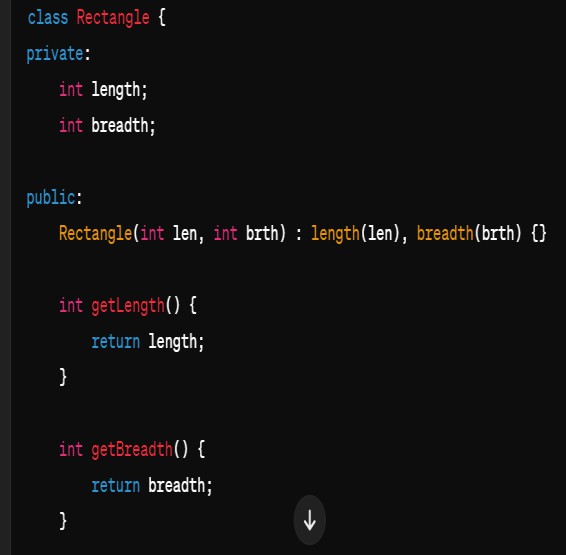


* Fungsi main() adalah titik masuk untuk eksekusi program.
* Objek Rect dari kelas Rectangle dibuat.
* Nilai width dan height objek Rect diatur masing-masing menjadi 5 dan 7 menggunakan metode setWidth() dan setHeight().
* Kemudian, luas objek Rect dicetak menggunakan metode getArea().

## Encapsulation



* **#include <iostream>** menyertakan library iostream yang diperlukan untuk operasi input-output.
* **using namespace std;** memungkinkan penggunaan istilah-istilah standar dari namespace std tanpa menyertakan std:: sebelumnya.



* Kelas Rectangle memiliki dua variabel anggota length dan breadth, yang merepresentasikan panjang dan lebar persegi panjang.
* **Konstruktor Rectangle(int len, int brth)** digunakan untuk menginisialisasi objek Rectangle dengan panjang dan lebar yang diberikan.
* Metode **getLength() dan getBreadth()** digunakan untuk mendapatkan nilai panjang dan lebar persegi panjang.
* Metode **getArea()** mengembalikan hasil perkalian panjang dengan lebar, yang merepresentasikan luas persegi panjang.
* Metode **setLength(int len) dan setBreadth(int brth)** digunakan untuk mengatur nilai panjang dan lebar persegi panjang. Mereka memeriksa apakah nilai yang diberikan tidak negatif sebelum mengubah nilai variabel.

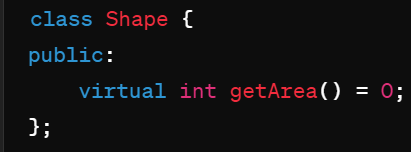


* **Fungsi main()** adalah titik masuk untuk eksekusi program.
* Objek rect dari kelas Rectangle dibuat dengan panjang 8 dan lebar 6.
* Luas persegi panjang dihitung dan dicetak menggunakan metode **getArea().**
* Metode **setLength(-5) dan setBreadth(-6)** dipanggil untuk menguji validasi input negatif. Pesan kesalahan dicetak karena panjang dan lebar tidak dapat negatif.
* Nilai panjang dan lebar akhirnya dicetak menggunakan metode **getLength() dan getBreadth().**

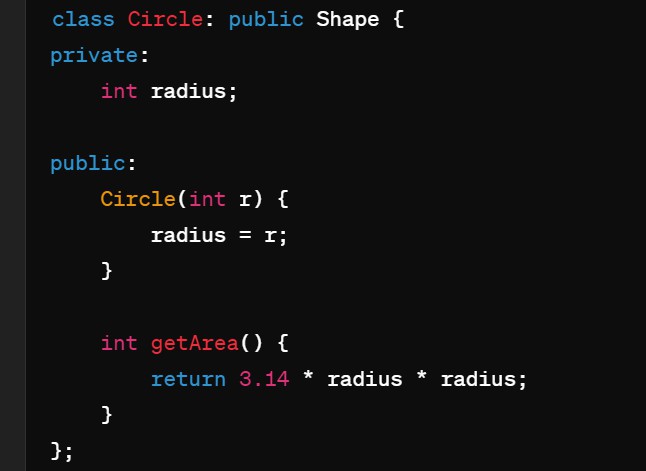
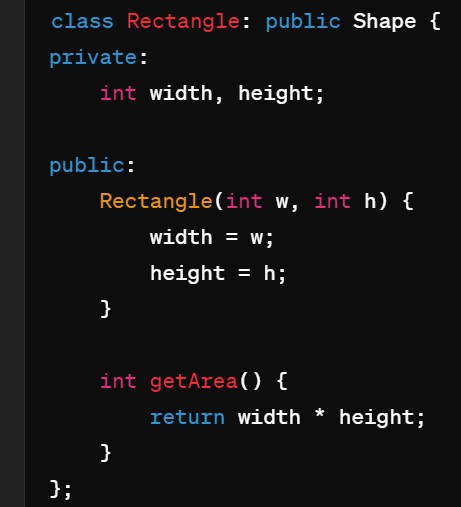
## Polimorphism

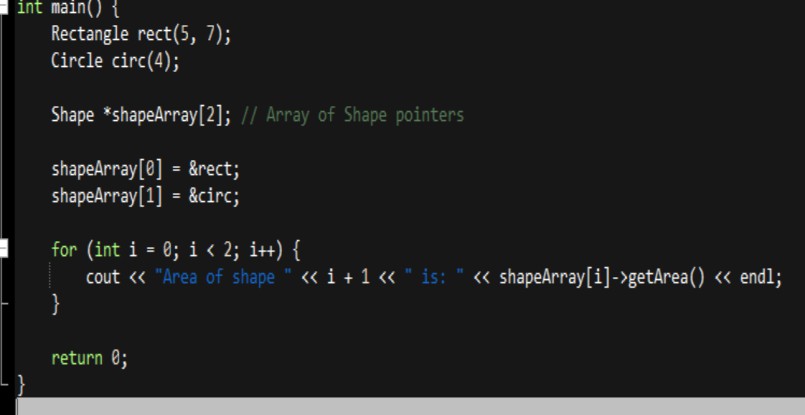


* Baris pertama **(#include <iostream>)** adalah preprocessor directive untuk menyertakan library iostream yang diperlukan untuk operasi input-output.
* **using namespace std;** memungkinkan penggunaan istilah-istilah standar dari namespace std tanpa menyertakan std:: sebelumnya.

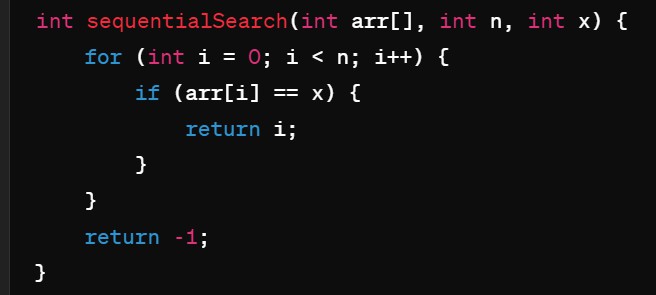


* Kelas Shape adalah kelas abstrak yang memiliki satu metode virtual murni **(getArea())**.
* Metode virtual murni dinyatakan dengan memberikannya nilai 0, yang menandakan bahwa kelas ini tidak dapat diinstansiasi (objeknya tidak dapat dibuat), dan kelas turunannya harus mengimplementasikan metode ini.

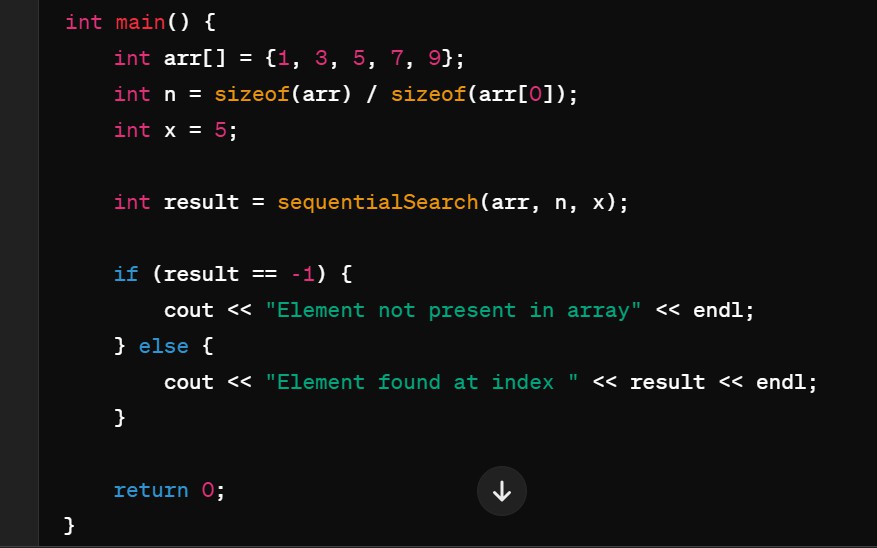


* Kelas Rectangle dan Circle masing-masing merupakan kelas turunan dari kelas Shape.
* Mereka mewarisi metode virtual murni **getArea() dari kelas Shape**.
* Masing-masing kelas memiliki konstruktor untuk menginisialisasi variabel anggota (**width dan height untuk Rectangle, dan radius untuk Circle).**
* Mereka mengimplementasikan metode getArea() untuk menghitung luas bentuk yang sesuai.
* Di dalam fungsi **main()**, dibuat objek rect dari kelas Rectangle dengan panjang 5 dan lebar 7, dan objek circ dari kelas Circle dengan jari-jari 4.
* Dideklarasikan sebuah array shapeArray dari pointer ke objek Shape.
* Pointer pertama menunjuk ke objek rect, dan pointer kedua menunjuk ke objek circ.
* Dilakukan iterasi melalui array **shapeArray** untuk mencetak luas dari setiap bentuk menggunakan metode **getArea()** yang dioverride oleh kelas turunannya.

# Pencarian (Searching)

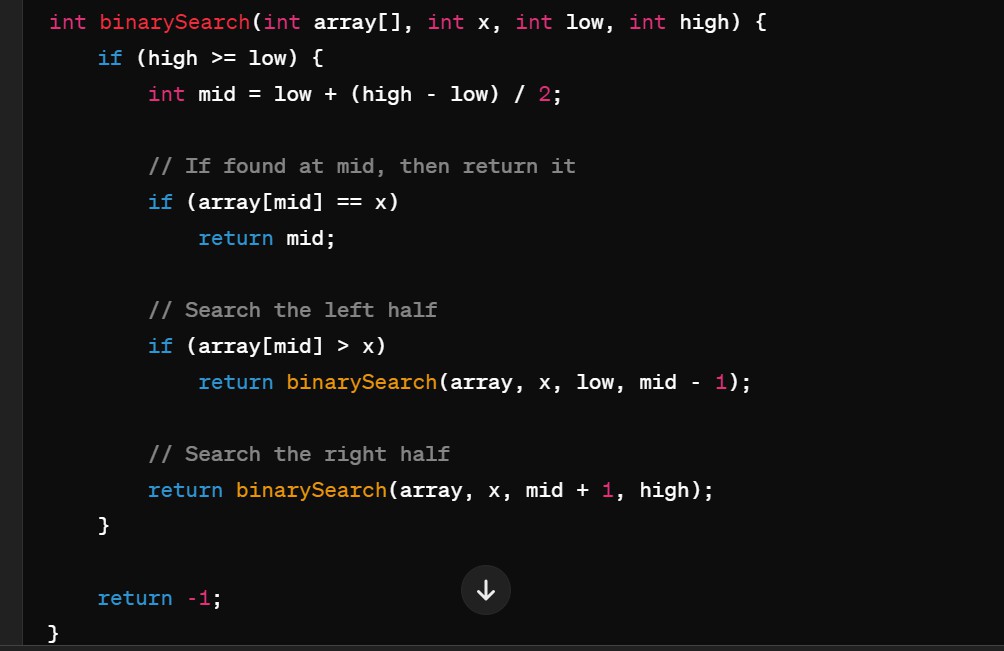
* 1. **Sequential Search**
* **#include <iostream>** adalah preprocessor directive yang menyertakan library iostream yang diperlukan untuk operasi input-output.
* **using namespace std;** memungkinkan penggunaan istilah-istilah standar dari namespace std tanpa menyertakan std:: sebelumnya.
* Fungsi **sequentialSearch()** menerima tiga parameter: array integer (arr), ukuran array (n), dan elemen yang dicari (x).
* Fungsi ini melakukan pencarian berurutan dalam array arr untuk menemukan elemen x.
* Iterasi dilakukan melalui seluruh elemen array, dan jika elemen yang dicari

(x) ditemukan, fungsi mengembalikan indeks di mana elemen tersebut ditemukan.

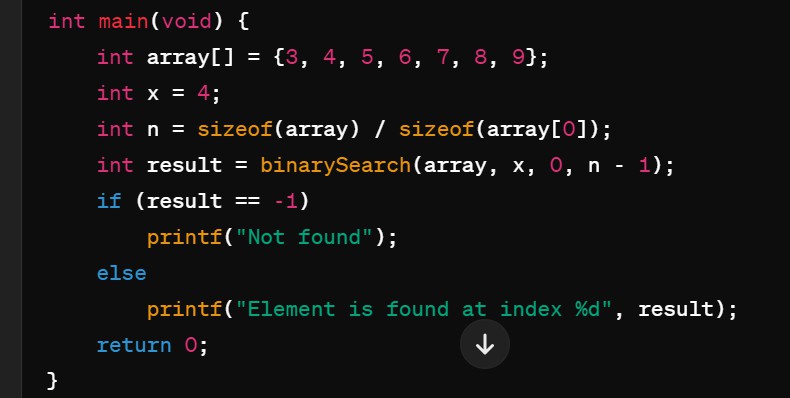
* Jika elemen tidak ditemukan, fungsi mengembalikan -1.
* Di dalam fungsi main(), sebuah array integer arr diinisialisasi dengan beberapa nilai.
* Variabel n dihitung sebagai ukuran array arr.
* Variabel x diberi nilai yang akan dicari dalam array.
* Fungsi sequentialSearch() dipanggil untuk mencari nilai x dalam array arr.
* Hasil pencarian disimpan dalam variabel result.
* Pesan yang sesuai dicetak tergantung pada apakah elemen ditemukan atau tidak.

## Binary Search

* **#include <iostream>** adalah preprocessor directive yang menyertakan library iostream yang diperlukan untuk operasi input-output.
* **using namespace std;** memungkinkan penggunaan istilah-istilah standar dari namespace std tanpa menyertakan std:: sebelumnya.



* Fungsi binarySearch() menerima empat parameter: **array integer (array), elemen yang dicari (x), indeks terendah (low), dan indeks tertinggi (high)** dalam pencarian saat ini.
* Fungsi ini menggunakan pendekatan rekursif untuk mencari elemen x dalam array array menggunakan pencarian biner.
* Pada setiap iterasi, fungsi menghitung mid (indeks tengah), dan membandingkan elemen tengah dengan elemen yang dicari (x).
* Jika elemen ditemukan, fungsi mengembalikan indeks di mana elemen tersebut ditemukan.
* Jika tidak ditemukan, fungsi memanggil dirinya sendiri untuk mencari di setengah array yang sesuai.



* Di dalam fungsi main(), sebuah array integer array diinisialisasi dengan beberapa nilai.
* Variabel x diberi nilai yang akan dicari dalam array.
* Variabel n dihitung sebagai ukuran array array.
* Fungsi binarySearch() dipanggil untuk mencari **nilai x dalam array, array menggunakan indeks dari 0 hingga n - 1.**
* Hasil pencarian disimpan dalam variabel result.
* Pesan yang sesuai dicetak tergantung pada apakah elemen ditemukan atau tidak.