Bahasa C++: Contoh Program Kecil

IF2210 – Semester II 2024/2025

RSP, IL

Hello World

Bahasa C

```c
/* File : helloworld.c */
/* Author : NIM - Nama */
#include "stdio.h"

int main(){
  printf("Hello World\n");
  return 0;
}
```

Bahasa C++ [versi prosedural]

```
/* File : helloworld.cpp */
/* Author : NIM - Nama */
#include <iostream>
using namespace std;
int main(){
  cout << "Hello World" << endl;
  return 0;
}
```

Bahasa C++ [versi OO]

```
/* File : hello.h */
/* Author : NIM – Nama */
class hello {
  public :
    hello();
};
```

——

```
/* File : hello.cpp */
/* Author : NIM - Nama */
#include "hello.h"
#include <iostream>
using namespace std;
hello::hello(){
  cout << "Hello World"
       << endl;
}
```

——

```
/* File:hellotest.cpp */
/* Author: NIM -Nama */
#include "hello.h"
#include <iostream>
using namespace std;

int main(){
  hello h; //ctor dihidupkan
  return 0;
}
```

——

Passing Parameter

```
// Z.h
#ifndef _Z_H
#define _Z_H
class Z {
  public:
    void print();          // print nilai val
    void print(int i);     // print nilai i
    void set(int x);       // set val dengan x
    int  add(int x, int y); // mengirimkan x+y
    void add(int x);       // menambah val dengan x
  private:
    int val;
};
#endif
```

```
// Z.cpp
#include "Z.h"
#include <iostream>
using namespace std;
void Z::print() { cout << "val=" << this->val << endl; }
void Z::print(int i) { cout << "i=" << i << endl; }
void Z::set(int x) {  this->val = x; }
int  Z::add(int x, int y) { return x+y; }
void Z::add(int x) {  this->val =  this->val + x; }
```

Perhatikan cara invokasi & passing parameter

```
#include <iostream>
#include "Z.h"
using namespace std;
int main() {
  Z z;
  z.set(2);
  z.print();
  cout << z.add(4,5) << endl;
  z.add(3);
  z.print();
  return 0;
}
```

Perhatikanlah bahwa :

1. Tidak ada ctor, tapi program dapat berjalan dengan baik
2. Prosedur bernama `print` dan fungsi bernama `add` ada dua, namun parameternya berbeda
3. Fungsi `int add(int x, int y)` tidak memakai nilai `val`.

ctor,cctor, dtor

```
// X.h
#ifndef _X_H
#define _X_H
class X {
  public:
    X();          // ctor
    X(int);       // ctor dengan parameter
    X(const X&);  // cctor
    ~X();         // dtor
    void print();  // prosedur untuk print atribut
  private:
    int x;        // atribut kelas
};
#endif
```

```
// X.cpp
#include "X.h"
#include <iostream>
using namespace std;
// Perhatikan bahwa umumnya tidak ada cout pada ctor, cctor, dtor.
// Pada contoh ini hanya dituliskan untuk menunjukkan kapan method dipanggil.
X::X() { x=0; cout << "ctor()" << endl; }
X::X(int a) { x=a; cout << "ctor(int)" << endl; }
X::X(const X& ox) { x=ox.x; cout << "cctor(X)" << endl; }
X::~X() { cout << "dtor()" << endl; } // pada kasus ini dtor does nothing
void X::print() { cout << "Nilai x=" << X::x << endl; }
```

```
// Y.h -- contoh kelas tanpa ctor, cctor, dtor
#ifndef _Y_H
#define _Y_H
class Y {
  public:
    void print();
  private:
    int y;
};
#endif
```

```
// Y.cpp
#include "Y.h"
#include <iostream>
using namespace std;
void Y::print() { cout << "Nilai y=" << Y::y << endl; }
```

```cpp
// mX.cpp
#include <iostream>
#include "X.h"
#include "Y.h"
using namespace std;
int main() {
  X x;
  X x1=x;  // karena ada cctor, maka bukan BITWISE COPY melainkan memanggil
           // cctor
  X* ptrx; // ptrx adalah pointer, harus di-new
  X* ptr1 = new X();
  x.print();
  x1.print();
  ptr1->print();

  Y oy; oy.print();
  Y y1 = oy; y1.print();
  Y* ptry = new Y(); ptry->print();

  return 0;
}
```

const

Tujuan topik ini

1. Mahasiswa memahami tentang sifat “immutability” dari data yang disimpan
2. Secara khusus, mengenai berbagai arti *keyword* const dalam bahasa C++ (melalui program kecil dalam dokumen ini dan dalam bahasa lain (lihat bacaan yang diberikan)

```
const int myconstant = 10;
   // An int which can't change value. similar to #define in C but better.
const int * myconstant;
   // Variable pointer to a constant integer.
int const * myconstant;
   // Same as above.
int * const myconstant;
   // Constant pointer to a variable integer.
int const * const myconstant;
   // Constant pointer to a constant integer.
void mymethod(QString const &myparameter);
   /* myparameter will not be altered by the method. & means it can be
      altered  but here we just want it to be used because it saves taking
      a copy */
```

How to initialize const member in C++?

```
// Contoh berikut salah, sebab Val adalah konstanta.
class Something {
  private:
    const int val;
  public:
    something() { val = 5; }
}
```

Lantas bagaimana? ctor initialization list

```
class Foo {
  private:
    const int data;
  public:
    Foo(int x): data(x) {...} // data diinisialisasi dengan x
};


class Bar: Foo {
  public:
    Bar(int x): Foo(x) {...}  // ctor Bar(int) diawali dengan
                              // memanggil ctor Foo(int)
};


class Baz {
  public:
    Foo* const foo;
    Baz(Foo* f): foo(f) {...} // foo diinisialisasi dengan f
};
```