Program Untuk menulis "Hello World.." ke layar

```
//hello.cpp

#include <iostream>
using namespace std;
int main() {
   cout << "Hello World.." << endl;
   return 0;
}
```

Program Untuk menulis "Hello World.." ke layar (versi -2)

```
//File : mHello.cpp
// main program yang tugasnya adalah menghidupkan sebuah objek aktif
// yaitu driver pengetest objek yang dilahirkan dari kelas Proses
#include "Proses.h"
using namespace std;

int main()
{
 Proses P;
 return 0;
}
```

```
/*File : Proses.h */
#ifndef _Proses_H
#define _Proses_H
class Proses {
public :
  Proses();             //ctor
};
#endif
```

```
/*File : Proses.cpp */
#include <iostream>
#include "Proses.h"
using namespace std;
// perhatikan bahwa pada umumnya tidak dituliskan "cout" pada ctor, cctor, dtor
// pada contoh ini dituliskan utk menunjukkan bahwa ada pengaktifan!
 Proses::Proses() { cout<<"Hello World.." << endl; };
```

Contoh Passing parameter

```
// Kelas Z dengan satu atribut val
#ifndef _Z_H
#define _Z_H
class Z {
public :
  void Print();             //Print nilai val
  void Print(int i);        //Print nilai i yang diberikan sbg parameter aktual
  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
```

```
// File Z.cpp
#include "Z.h"
#include <iostream>
using namespace std;

void Z::Print() { cout << "val= "<<  Z::val << endl;} //Print nilai val

void Z::Print(int i)   //Print nilai i yang diberikan sbg parameter aktual
  { cout << Z::val <<" +"  << i << "=" << Z::val + i << endl; }

void Z::Set (int x) { Z::val = x; } // set val dengan x

int Z::Add (int x, int y) // mengirimkan x+y
  { return x+y; }

void Z::Add (int x)  { Z::val = Z::val + x;}  // menambah val dengan x
```

Perhatikan cara melakukan invokasi dan 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.

Contoh ctor, cctor, dtor

```
/*File : 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();  // proeedur, untuk memprint atribut

private :
  int x;        //atribut kelas

};
#endif
```

```
/*File : X.cpp */
#include <iostream>
#include "X.h"
using namespace std;
// perhatikan bahwa pada umumnya tidak dituliskan "cout" pada ctor, cctor, dtor
// pada contoh ini dituliskan utk menunjukkan bahwa ada pengaktifan
 X::X() {x=0; cout<<"ctor X().." << endl; };             //ctor
 X::X(int a) {x=a; cout << "ctor X(int).."<< endl;  }  // ctor dg parameter
 X::X(const X& OX){x = OX.x; cout<< "cctorX.." << endl;}   //cctor
 X::~X(){cout << "dtor.." <<endl;}          //dtor, kasus ini tak perlu

  void X::Print(){
      cout <<" Nilai x=: "<< x<<endl;}  // proeedur, untuk memprint atribut
```

```
/*File : Y.h */
/* kelas tanpa ctor, cctor, dtor*/
#ifndef _Y_H
#define _Y_H
class Y {
public :
  void Print();  // proeedur, untuk memprint atribut

private :
  int y;        //atribut kelas

};
#endif
```

```
/*File : Y.cpp */
#include <iostream>
#include "Y.h"
using namespace std;
  void Y::Print(){
      cout <<" Nilai y=: " << y <<endl;}  // proeedur, untuk memprint atribut
```

```
/*File : mX.cpp */
#include <iostream>
#include "X.h"
#include "Y.h"
using namespace std;
int main() {

  X x;         //x adalah seperti "struct" pada bhs C
  X x1=x;      // karena ada cctor maka bukan bitwise copy, tapi cctor dijalankan
  X* ptrx;     //ptrx adalah pointer; hrs 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;
}
```

Output program : perhatikan bahwa tanpa ctor, bagaimana menginisialisasi atribut ?
Itulah sebabnya diperlukan ctor
Ctor diperlukan untuk menginisialisasi nilai atribut saat objek diciptakan

```
ctor X()..
cctorX..
ctor X()..
 Nilai x=: 0
 Nilai x=: 0
 Nilai x=: 0
 Nilai y=: 134515842
 Nilai y=: 134515842
 Nilai y=: 0
dtor..
dtor..
```

JAVA

Program Untuk menulis "Hello World!!" ke layar

(versi-1 : hanya main program)

```
// File : hello.java
// nama file harus sama dengan nama kelas
class hello {
public static void main(String args[]){
     System.out.println("Hello World!!");
  }
}
```

Program Untuk menulis "Hello World!!" ke layar

(versi-2 : dengan main program yang menciptakan objek yang menulis "Hello World!!" saat diciptakan)

```
// File : SayHello.java
class SayHello {
// ctor
   SayHello () {
     System.out.println("Hello World!!");
  }
}
```

```
//  File : mhello.java
class mhello {
 public static void main(String args[]){
      SayHello S= new SayHello();
        }
}
```

Contoh Pemakaian Kelas dan passing parameter

```
// File : Z.java
class Z {
private int val;

public void Print() { System.out.println ( "val= " +  val );} //Print nilai val
public void Print(int i)   //Print nilai i yang diberikan sbg parameter aktual
  { System.out.println (""+ val + " +"  + i + "=" + val + i ); }

public void Set (int x) { val = x; } // set val dengan x

public int Add (int x, int y) // mengirimkan x+y
  { return x+y; }

public void Add (int x)  { val = val + x;}  // menambah val dengan x
}
```

```
// File : mZ.java
class mZ {
public static void main(String args[]){
  Z z= new Z();
  z.Set(2);
  z.Print();
  System.out.println( z.Add (4,5) ) ;
  z.Add (3);
  z.Print();
}
}
```

Contoh ctor

```
/* File : X.java */
public class X {
 private int x;
 public X() {x=0;}                    //ctor
 public X(int a) {x = a; };           // ctor dg parameter
 public void Print() { System.out.println("nilai x= "+x); }

}
```

```
/* file : Y.java */
/* pada kelas ini tidak dituliskan ctor */
public class Y {
 private int y;
 public void Print() { System.out.println("nilai y = "+ y); }

}
```

```
/* file : MX.java*/
public class MX {
 public static void main(String args[]){
 // X x;         //x adalah seperti "struct" pada bhs C
 // X x1=x;      // karena ada cctor maka bukan bitwise copy, tapi cctor dijalankan
  X ptrx;    //ptrx adalah pointer; hrs di-new
  X ptr1=new X(); ptr1.Print();
//  x.Print()
//  x1.Print();
//  ptr1->Print();

 Y pty=new Y(); pty.Print();
}
 }
```

Output :
You are lucky, y=0. But ..... ???

```
nilai x= 0
nilai y = 0
```