



PEMROGRAMAN LANJUT

03 | Constructor in C++

Analogi

- Setiap pagi pergi ke kantor
- Sebelum berangkat selalu menyiapkan:



Sarapan



Baju Kerja



Kendaraan

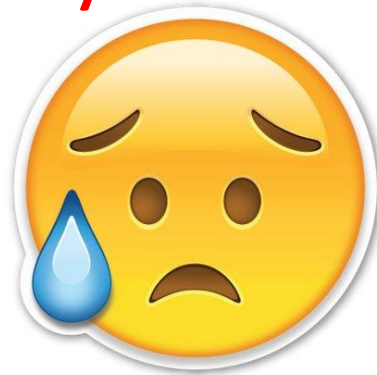
Analogi

#Sarapan

#Baju Kerja

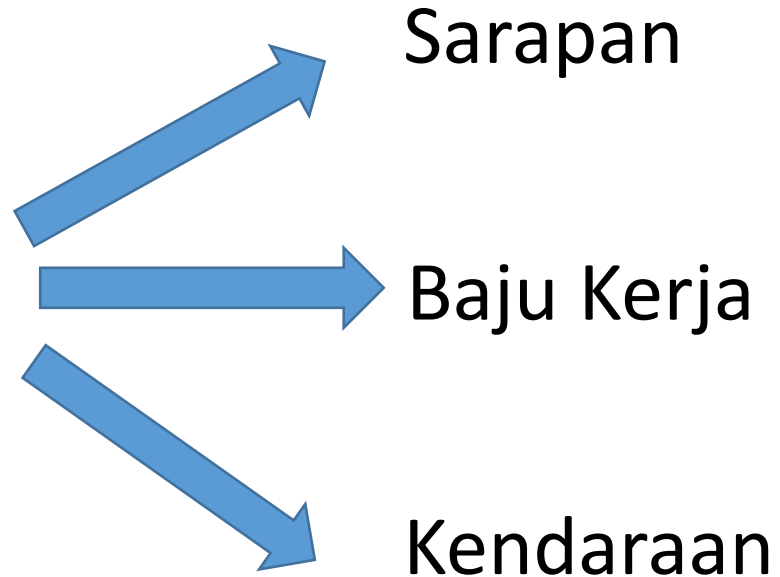
#Kendaraan

Small tasks consume lot of time
every morning



Bagaimana bila kita memrogram robot untuk
melakukan aktifitas tersebut?

Analogi

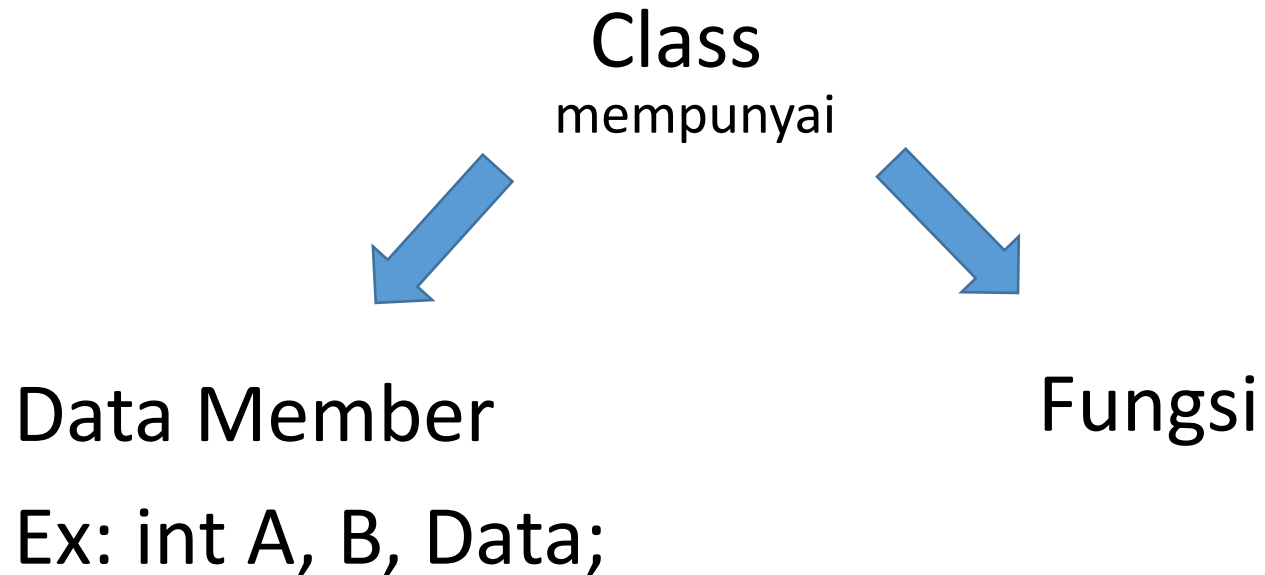


Sekarang kita bisa menggunakan waktu lebih produktif



Konstruktör

Apa yang kita punya?



A **constructor** is a special kind of class member function that is automatically called when an object of that class is instantiated.

Contoh

```
class demo
{
int a, b, data;
public:
demo()
{
a=5;
b=15; #Menginisiasi nilai
data=25
};
```

#Ini adalah konstruktor

#Konstruktor mempunyai nama yang sama dengan nama kelas

a=5;

b=15; #Menginisiasi nilai

data=25

#Konstruktor ini akan dipanggil saat objek dibuat

#Yang artinya akan dipanggil secara otomatis dan a, b, data akan berisi 5, 15, 25 didalamnya secara default

Seperti robot



Silahkan Dicoba

```
1  #include <iostream>
2  using namespace std;
3
4  class demo
5  {
6      int data;
7      public:
8          demo()
9          {
10             cout<<"Ngoding emang EZ..."<<endl;
11         };
12     };
13
14     int main()
15     {
16         demo obj;
17         return 0;
18     }
```

#Tambahkan objek lain seperti qwe,
asd, zxc
#Lihat apa yang terjadi

Kesimpulan

- Konstruktor di deklarasikan di public
- Konstruktor akan dipanggil ketika objek dibuat
- Konstruktor tidak mempunyai return type bahkan void

Tipe Konstruktor



Tidak
mempunyai
parameter

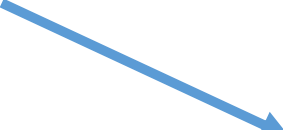


Mempunyai
parameter

Konstruktor Tidak Mempunyai Parameter

- Konstruktor yang tidak mempunyai nilai parameter (pass)
- Contohnya adalah Default Constructor
 - Setiap Program pasti/harus mempunyai konstruktor
 - Namun program akan tetap berjalan meskipun kita tidak membuat konstruktor. Mengapa?
 - Secara default, compiler akan memberi konstruktor dengan body yang kosong

Don't pass any parameter



```
1  class_name()  
2  {  
3  //body  
4  }
```

Konstruktor Berparameter

- Konstruktor yang mempunyai nilai parameter (passing value)
- Contohnya
 - Memberi nilai pada variabel data
 - Dikarenakan konstruktor dipanggil secara otomatis, darimana datangnya nilai tersebut?
 - Saat pembuatan object
 - Namun program ini akan error ketika membuat objek lainnya. Mengapa?
 - Karena Setiap program harus/pasti mempunyai default constructor

```
class demo
{
    int a, b, data;
    public:
    demo(int a)
    {
        data=a;
    }
};

Int main()
{
    demo obj1 (55);
    return 0;
}
```

Konstruktor Berparameter

Tidaklah penting sebuah objek memiliki nilai kembalian (passing value) atau tidak. Contohnya

```
demo obj1(50), obj2;
```

Untuk ini kita harus membuat default konstruktornya. Mengapa?

Ketika kita membuat konstruktor, compiler tidak membuat default constructor

Silahkan Dicoba

```
1  #include <iostream>
2  using namespace std;
3
4  class demo
5  {
6      int data;
7      public:
8          demo(int a)
9          {
10             data=a
11          }
12          void display()
13          {
14             cout<<"data = "<<data<<endl;
15          }
16  };
17
18  int main()
19  {
20      demo obj(50),obj2;
21      obj.display();
22      return 0;
23  }
```

#Tambahkan pada public
demo()

{}

#Lihat apa yang terjadi

Silahkan Dicoba

```
1  #include <iostream>
2  using namespace std;
3
4  class demo
5  {
6      int data;
7      public:
8          demo(int a);
9          demo()
10         {
11
12         }
13         void display()
14         {
15             cout<<"data = "<<data<<endl;
16         }
17     };
18
19     demo::demo(int a)
20     {
21         data=a;
22     }
23
24     int main()
25     {
26         demo obj(50),obj2;
27         obj.display();
28         return 0;
29     }
```

//Deklarasi

//Definisi

#Kita juga bs meletakkan constructor diluar kelas

Ada Pertanyaan?

