Bahasa Pemrograman :: Object Oriented Programming

Julio Adisantoso
ILKOM IPB

19 April 2010

Outline

Contoh Program C++

```
#include <iostream>
using namespace std;

struct Time {
   int hour; // 0-23
   int minute; // 0-59
   int second; // 0-59
};
typedef struct Time TIME;

TIME t;
```

Prosedur dalam C++

```
void setTime(int h=0, int m=0, int s=0)
{  t.hour=h; t.minute=m; t.second=s;  }

void print(void)
{
   cout << endl << t.hour << ":" << t.minute
        << ":" << t.second << endl;
}

main() {
   setTime(10,20,30); print();
   setTime(10,20); print();
   return 0;
}
```

Object Oriented Programming

- Sering disingkat menjadi OOP
- Pada contoh sebelumnya, program memiliki data (time) dan beberapa prosedur yang terpisah
- ———
- Prinsip dasar OPP adalah membungkus prosedur dan data dalam satu obyek - **Encapsulation**
- OOP memodelkan obyek yang ada di dunia nyata ke dalam software obyek dalam pemrograman
- Implementasi dalam bentuk **Class**
- Berfungsi sebagai ADT (**Abstract Data Type**)
- Dalam kuliah ini, digunakan Java sebagai platform OOP

Mengapa Java

- Sederhana (Simple)
  - Bahasa pemrograman Java menggunakan sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit.
- Berorientasi obyek (Object Oriented)
- Terdistribusi (Distributed)
  - Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya libraries networking yang terintegrasi pada Java.
- Interpreted
  - Program Java dijalankan menggunakan interpreter yaitu Java Virtual Machine (JVM).
  - Hal ini menyebabkan source code Java yang telah dikompilasi menjadi Java bytecodes dapat dijalankan pada platform yang berbeda-beda.

Java

- Free for download ⇒ http://java.sun.com
- Unit terkecil program Java adalah Class yang terdiri dari methods (C: procedure) dan instance (C: data)
- Tahapan:
  - Edit - welcome.java
  - Compile - javac welcome.java
  - Load - .class
  - Verify
  - Execute - java welcome
- Contoh:

```
public class hello {
    public static void main(String[] args) {
        //Menampilkan "Hello world" dilayar
        System.out.println("Hello world!");
    }
}
```

UTS

1. Perhatikan program fungsional PLT-Scheme berikut:

```
(define (what n x)
  (when (> n 0)
     (if (= n 1) empty
        (cons (first x) (what (- n 1) (rest x)))
     )
  )
)
```

   a) Apa output program tersebut jika dipanggil fungsi berikut.
   (what 3 (list 10 20 30 40))
   b) Apa output program tersebut jika dipanggil fungsi berikut.
   (what 0 (list 10 20 30 40))
   c) Terjemahkan program tersebut ke dalam pemrograman logika GNU-Prolog

UTS

2. Buat program fungsional dan logika untuk menghitung pembagian a dan b-c. Program tidak menghasilkan apa-apa jika nilai pembagi adalah 0

   a) PLT-Sceme

```
> (bagi 20 7 7) #menghitung 20/(7-7)
> (bagi 20 12 4) #menghitung 20/(12-4)
2 1/2
```

   b) GNU-Prolog

```
?- bagi(20,7,7,X).
no
?- bagi(20,12,4,X).
X=2.5
yes
```

UTS

3. Buat program untuk menggabungkan list x dan y menjadi list z, dimana elemen x dan y diambil secara bergantian selama elemen list tersebut masih ada (lihat contoh).

   a) PLT-Sceme

```
> (gabung (list 1 2 3 4 5) (list 10 20))
(1 10 2 20 3 4 5)
> (gabung (list 1 2) (list 20 30 40 50 60))
(1 20 2 30 40 50 60)
```

   b) GNU-Prolog

```
?- gabung([1,2,3,4,5], [10,20], Z).
Z=[1,10,2,20,3,4,5]
yes
?- gabung([1,2], [20,30,40,50,60], Z).
Z=[1,20,2,30,40,50,60]
yes
```

UTS

4. Buat program memeriksa apakah suatu list merupakan prefiks dari list lainnya. List x merupakan prefiks dari list y jika semua elemen dari list x merupakan bagian paling depan dari list y (lihat contoh). List kosong merupakan prefiks dari list apapun. Anda tidak diperbolehkan menggunakan fungsi built-in prefix dalam PLT-Scheme.

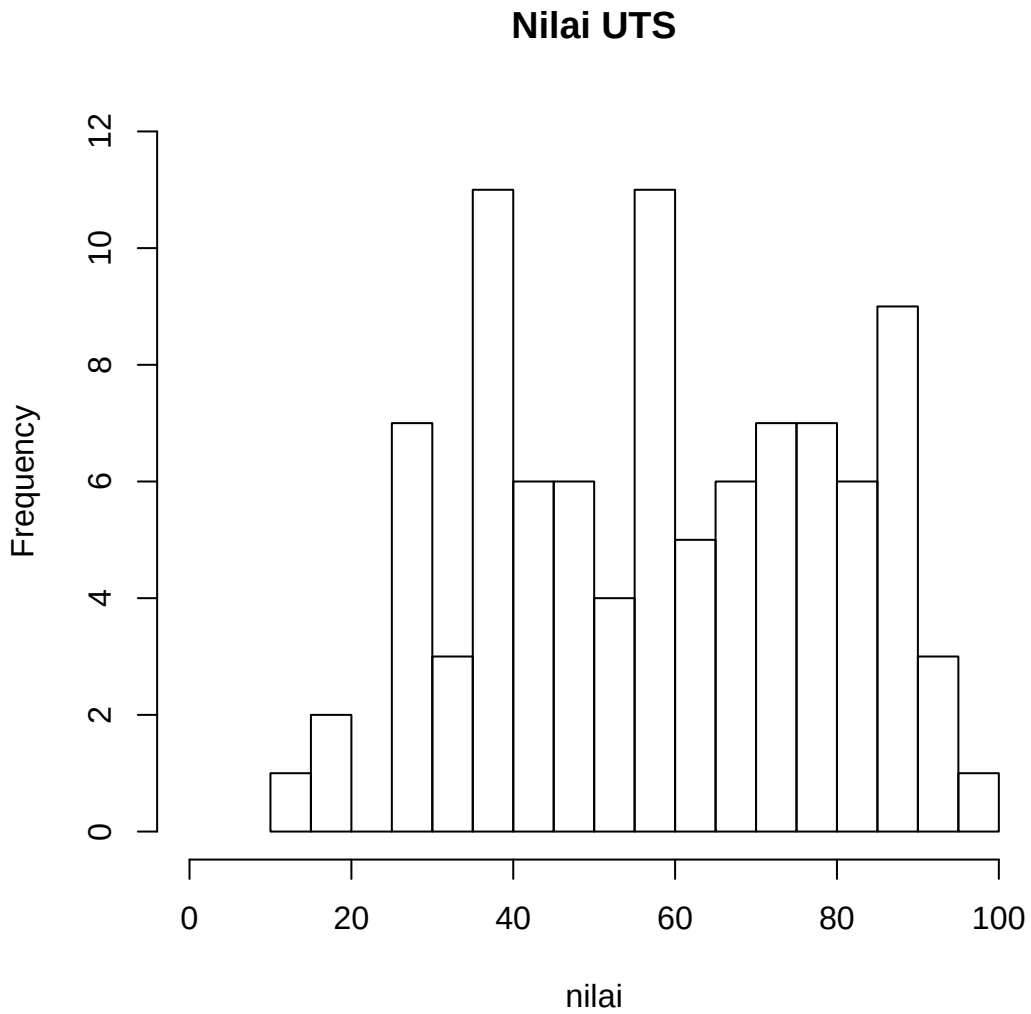
   a) PLT-Sceme

```
> (prefiks (list 1 2 3) (list 1 2 3 4 5 6))
#t
> (prefiks (list 1 2 3) (list 1 2 4 3 5 6))
#f
```

   b) GNU-Prolog

```
?- prefix([1, 2, 3], [1, 2, 3, 4, 5, 6]).
yes
?- prefix([1, 2, 3], [1, 2, 4, 3, 5, 6]).
no
```

Sebaran Nilai UTS



Sebaran Nilai UTS

```
1 |  38
2 |  067899
3 |  00135677888889
4 |  00111244667778
5 |  3555666778999999
6 |  1222467799
7 |  011111456678 9
8 |  00222234778 88
9 |  00001338
```