PEMROGRAMAN BERORIENTASI OBJEK

TEMPLATE

Rosa Ariani Sukamto

ROSA ARIANI SUKAMTO

Blog: http://hariiniadalahhadiah.wordpress.com

Facebook: https://www.facebook.com/rosa.ariani.sukamto

Email: rosa_if_itb_01@yahoo.com

TEMPLATE

Pendefinisian nama metode yang sama dengan parameter yang berbeda (*overloading*) pada bahasa pemrogramann C++ dapat dimudahkan penulisannya pada kode program dengan menggunakan kata kunci *template*

Template hanya berlaku untuk metode yang memiliki isi atau badan program yang sama persis, hanya berbeda tipe data parameter masukkannya

Hanya pada bahasa pemrograman C++

IMPLEMENTASI TEMPLATE - FUNGSI/PROSEDUR

Mesin.cpp

```
void tulisMasukan(int n){
  cout << "masukan : " << n <<
   endl;
}

void tulisMasukan(double n){
  cout << "masukan : " << n <<
   endl;
}

void tulisMasukan(string n){
  cout << "masukan : " << n <<
   endl;
}
```

```
template <class Masukan> void
   tulisMasukan(Masukan n){
  cout << "masukan : " << n <<
   endl;
}
```

IMPLEMENTASI TEMPLATE - FUNGSI/PROSEDUR LEBIH DARI SATU TIPE

Mesin.cpp

```
int keluaran(int n, double m){
  cout << "keluaran 2 tipe" << n
   << m << endl;
  return n;
}

char keluaran(char n, int m){
  cout << "keluaran 2 tipe" << n
   << m << endl;
  return n;
}

double keluaran(double n, char m){
  cout << "keluaran 2 tipe" << n
   << m << endl;
  return n;
}
```

```
template <class A, class B>
A keluaran(A n, B m){
  cout << "keluaran 2 tipe" << n
   << m << endl;
  return n;
}
```

IMPLEMENTASI TEMPLATE - OVERLOADING

Mesin.cpp

```
template <class T>
T keluaranO(T n){
  cout << "keluaran overloading" << n << endl;
  return n;
}

template <class T>
T keluaranO(T x, T y){
  cout << "keluaran overloading" << x << y <<
   endl;
   return (x * y);
}
```

IMPLEMENTASI TEMPLATE - KELAS (1)

Titik.h

```
template <class T>
class Titik{
  private:
    int x; // koordinat x
    int y; // koordinat y

  public:
    Titik();
    Titik(int xp, int yp);
    int getX();
    void setX(int xp);
    int getY();
    void setY(int yp);
    ~Titik();
};
```

Titik.cpp

```
#include "Titik.h"
template <class T>
Titik<T>::Titik(){
  /*konstruktor*/
  Titik<T>::x = 0;
  Titik<T>::y = 0;
}

template <class T>
Titik<T>::Titik(int xp, int yp){
  /*konstruktor*/
  Titik<T>::x = xp;
  Titik<T>::y = yp;
}
```

IMPLEMENTASI TEMPLATE - KELAS (2)

Titik.cpp

```
template <class T>
int Titik<T>::getX(){
  /*mengembalikan nilai x*/
  return Titik<T>::x;
}

template <class T>
void Titik<T>::setX(int xp){
  /*mengeset nilai x*/
  Titik<T>::x = xp;
}

template <class T>
int Titik<T>::getY(){
  /*mengembalikan nilai y*/
  return Titik<T>::y;
}
```

```
template <class T>
void Titik<T>::setY(int yp){
  /*mengeset nilai y*/
  Titik<T>::y = yp;
}

template <class T>
Titik<T>::~Titik(){
}
```

IMPLEMENTASI TEMPLATE - KELAS (3)

Tulis.cpp

```
template <class T>
class Tulis{

 public:
   Tulis(T kata){
     /*konstruktor*/
     cout << "isi kata masukan : " << kata << endl;
   }
};
```

IMPLEMENTASI TEMPLATE - MAIN

```
#include <cstdio>
#include <iostream>
using namespace std;
#include "mesin.cpp"
#include "Titik.cpp"
#include "Tulis.cpp"
int main(){


  Titik<int> t1(28, 1);
  string kata = "membahas
   template";


  tulisMasukan(18);
  tulisMasukan(28.11);
  tulisMasukan("prosedur
  tulisMasukan dengan masukan
   string");
```

```
  keluaran(11, 11.82);
  keluaran(9, 'A');
  keluaran(9.81, 'A');
  keluaranO('A');
  keluaranO(81);
  keluaranO(82.81);
  keluaranO(3, 5);
  keluaranO(18.9, 28.11);


   cout << "t1 : x : " <<
   t1.getX() << "  y : " <<
   t1.getY() << endl;
  Tulis<string> t(kata);


  return 0;


}
```

TEMPLATE PADA LIBRARY C++ - STL (STANDARD TEMPLATE LIBRARY) VECTOR (1)

```
#include <vector>
#include <cstdio>
#include <algorithm>
#include <iostream>

using namespace std;

int main(){

   vector<string> vc;

   vc.push_back("merah");
   vc.push_back("kuning");
   vc.push_back("hijau");
   vc.push_back("biru");
   vc.push_back("jingga");

   sort(vc.begin(), vc.end());

   int i;

   do{
        for(i=0;i<vc.size();i++){
                printf("%s ",
   vc[i].c_str());
        }
        printf("\n");
   }while(next_permutation(vc.begi
   n(), vc.end()));

   return 0;
}
```

TEMPLATE PADA LIBRARY C++ - STL (STANDARD TEMPLATE LIBRARY) VECTOR (2)

```
#include <vector>
#include <cstdio>
#include <algorithm>
#include <iostream>

using namespace std;

int main(){

   vector<int> v;

     v.push_back(6);
     v.push_back(2);
     v.push_back(7);
     v.push_back(4);
     v.push_back(9);
```

```
   sort(vc.begin(), vc.end());

   int i;

   do{
        for(i=0;i<vc.size();i++){
                printf("%d ",
   vc[i]);
        }
        printf("\n");
   }while(next_permutation(vc.begi
   n(), vc.end()));

   return 0;
}
```

TEMPLATE PADA LIBRARY C++ - STL (STANDARD TEMPLATE LIBRARY) SET (ISI UNIK) (1)

```
#include <cstdio>
#include <vector>
#include <set>
#include <string>
using namespace std;

int main (){
  //string
  set<string> s;

  s.insert("Hello");
  s.insert("World");
  s.insert("Hello");
  s.insert("World");

  set<string>::const_iterator it;
```

```
  vector<string> str;


  //mending dipindah ke vector
   kalau memang sudah unik di set
  int i = 0;
  for (it =
   s.begin();it!=s.end();it++) {
   str.push_back("" + *it);
   printf("%s\n", str[i].c_str());
   i++;
  }
```

TEMPLATE PADA LIBRARY C++ - STL (STANDARD TEMPLATE LIBRARY) SET (ISI UNIK) (2)

```
#include <cstdio>
#include <vector>
#include <set>
using namespace std;

int main (){
  //integer
  set<int> arr;
  arr.insert(7);
  arr.insert(8);
  arr.insert(7);
  arr.insert(8);

  set<int>::const_iterator itn;

  vector<int> varr;
```

```
//mending dipindah ke vector kalau
   memang sudah unik di set
  i = 0;
  for (itn =
   arr.begin();itn!=arr.end();itn+
   +) {
   varr.push_back(0 + *itn);
   printf("%d\n", varr[i]);
   i++;
  }

  return 0;
}
```

TEMPLATE PADA LIBRARY C++ - STL (STANDARD TEMPLATE LIBRARY) PAIR

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

int main (){
  int n;

  scanf("%d", &n);
  pair<int, int> data[n];
  int i;
  for(i=0;i<n;i++){
   scanf("%d %d", &data[i].first,
   &data[i].second);
  }
```

```
  printf("isi dari pair\n");

  for(i=0;i<n;i++){
   printf("%d %d\n",
   data[i].first, data[i].second);
  }

  return 0;
}
```

CONTOH SOAL

Buatlah kelas template dengan nama kelas OperasiAritmetika yang memiliki atribut a dan b (a dan b dapat diisi tipe apapun yang termasuk angka) buatlah metode tambah, bagi, kali, kurang yang mengoperasikan kedua atribut untuk dikenai operasi aritmetika (+, /, x, -)

DAFTAR PUSTAKA

S, Rosa A. dan M. Shalahuddin. 2011. Modul Pembelajaran: Pemrograman Berorientasi Objek. Modula: Bandung.