Tifany Inne / 1701295104 / 02PPT / 37

UAS T0044 PAGI 20xx

1. Pilihan (20 poin)
2. Opsi berikut yang *bukan* merupakan fitur dari OOPL adalah

a. encapsulation b. inheritance c. polymrophism **d. template**

1. Secara sederhana *polymorphism* dapat didefinisikan sebagai *multiple routine with the same name.* Opsi berikut yang *tidak dapat digunakan* untuk mendefinisikan polymorphism adalah

a. early binding **b. late binding** c. overriding d. overloading

1. Dua prototipe berikut : *add (const Base &)* dan *add (const Derived &)* berbeda dalam hal:

**a. scope** b. lifetime c. signatures d. precedence

1. Meskipun mendukung penggunaan *multiple inheritance* tetapi penggunaan fitur ini tidak dianjurkan di C++. Opsi berikut yang dianjurkan sebagai penggantinya adalah

**a. composition**  b. template

c. polymorphism d. overloading function

1. Selain sebagai *tool* untuk membentuk generic programming, t*emplate* juga merupakan *tool* untuk membentuk

a. universal polymorphism **b. static polymorphism**

c. ad hoc polymorphism d. subtype polymorphism.

6. Paradigma-paradigma berikut yang mendukung pemrograman C++ adalah

a. procedural dan object oriented programming

b. procedural dan generic programming

c. generic dan object oriented programming

**d. procedural, object oriented dan generic programming.**

7. Ide dari *generic programming* adalah mengganti

**a. static function** b. inline function

c. virtual function d. friend function

dengan fungsi yang ada di template parameter.

8. Iterator tidak lain adalah

a. generalized reference **b. generalized pointer**

c. specialized reference d. specialized pointer.

9. *Tipe inheritance* yang digunakan untuk melakukan *encapsulation* adalah

a. private b. protected

**c. public** d. private, protected dan public

1. Opsi berikut yang *bukan* merupakan *karakteristik* dari fungsi virtual adalah

a. dapat di override di derived class

b. digunakan untuk mendukung polimorfisme

c. deklarasi virtual di base class

**d. deklarasi virtual di derived class**

1. Uraian (40 poin)

1. (10 poin) Uraikan mengapa *constructor* dan *destructor* tidak diwariskan di C++ meskipun mereka kita deklarasikan dengan *level akses public* atau *protected*. Uraikan juga mengapa *fungsi friend* tidak diwariskan di klas turunan. Selanjutnya, sebutkan pula sejumlah fitur lain di klas induk yang tidak diwariskan di klas turunan.

Jawab:

*Karena constructor bukan interface dari class, sementara yang diwariskan adalah interface dari class. Destructor tidak diwariskan karena satu class hanya boleh mempunyai satu destructor.*

*Friend function tidak diwariskan karena friend function membuat data yang ada di class itu dapat di akses oleh fungsi tertentu. Saat friend diwariskan, datanya akan berubah dan friend function itu harus berubah pula.*

*Fitur lain yang tidak diwariskan adalah assigment operator.*

2. (10 poin) Konsep *inheritance* memunculkan fitur baru dalam C++ yaitu *reuseability*, *dominating* dan *overriding*. Jelaskan melalui ***penggalan kode***, apa yang kamu ketahui tentang ketiga fitur tersebut.

**Jawab:**

*Reusability berarti penggunaan ulang member yang ada di class parent. Misalnya class A:*

*class A{*

*private:*

*int x;*

*…};*

*bila class B mewarisi class A, maka int x ada di class B pula*

*Dominating berarti beberapa fungsi dengan nama yang sama akan di dominasi oleh satu fungsi dan yang lainnya di tutup. Misal:*

*void f(double, double); // at global scope*

*struct Grandparent {*

*void f(int);*

*void f(double, double);*

*};*

*struct Parent : public Grandparent {*

*void f(int); // hides all overloads of Grandparent::f*

*};*

*struct Child : public Parent {*

*void g() { f(2.14, 3.17); } // resolves to Parent::f*

*};*

*fungsi f di global scope ditutup oleh fungsi f dari grandparent yang ditutup fungsi f parent.*

*Overriding berarti suatu fungsi di class anaknya menimpa fungsi di class parent. Misalnya class A:*

*class A{*

*…*

*void X(...){...}*

*…};*

*bila class B mewarisi class A dan membuat fungsi X, maka fungsi yang akan digunakan object B adalah fungsi dari class B.*

3. (10 poin) Ada banyak tipe polymorphism yang kita kenal diantaranya adalah *parametric polymorphism*, *ad hoc polymorphism*, *subtype polymorphism,* *override polymorphism* dan sebagainya. Uraikan apa yang kamu ketahui tentang tipe-tipe polymorphism tersebut.

**Jawab:**

*parametric polymorphism*: *kode yang ditulis tanpa menulis secara spesifik tipenya sehingga dapat digunakan dalam berbagai tipe.*

*ad hoc polymorphism: fungsi dengan nama sama tapi parameter berbeda sehingga satu nama berarti banyak.*

*subtype polymorphism: fungsi yang diturunkan tetapi isi diganti*

4. (10 poin) Banyak penulis buku menyatakan bahwa untuk *class* X paling sedikit ia

mempunyai satu *constructor* yaitu *default constructor* dengan prototype X(). Tetapi

John Hubbard dalam bukunya *Programming with C++* menyatakan bahwa setiap

*class* X paling sedikit mempunyai dua *constructor* yaitu *default* X() dan *copy*

*constructor* yang prototipe sintaksnya adalah *X (const X & )*, fungsi anggota dengan

satu parameter dan *pass by constant reference*.

a. Kapan *copy constructor* dieksekusi?

b. ***Tulislah fungsi main()*** untuk menguji *copy constructor* yang ada dalam penggalan

kode berikut:

class Rational {

public : Rational(int n, int d) : num(n), den (d) { }

Rational(const Rational & r) : num(r.num), den(r.den)

{ cout << “COPY CONSTRUCTOR CALLED\n”;}

private: int num, den;

};

Rational f(Rational r) {

Rational s = r;

return s;

}

**Jawab:**

1. *Saat menginisialisasi suatu objek dan mengisinya dengan isi dari objek lain, dan saat mengirim objek by value.*
2. *Int main(){*

*Rational a;*

*Rational b=a;*

*Return 0;*

*}*

1. Pemrograman (40 poin)

a. (10 poin) Desain *class String* dengan dua data anggota *len* bertipe *unsigned* untuk mewakili panjang string dan *buf*  bertipe *pointer ke character* untuk menampung *string* karakterr aktual. Untuk menyederhanakan kasus, fungsi anggota dalam class ini cukup tiga buah yaitu *default constrctor* dengan prototipe *String (unsigned = 0)*, *copy constructor* dengan prototype *String (const String&)* dan *destructor* dengan prototype *~String().*

b. (10 poin) Tulis implementasi ketiga fungsi anggota tersebut di luar class.

c. Tulislah *the bubble sort (function) templates* yang dapat digunakan untuk men-sort *any data type* termasuk *class String* di atas. Program yang Anda desain paling tidak harus terdiri dari tiga komponen yaitu

* (10 poin) *function template sortBubble()* di mana type parameter T nantinya akan diganti dengan tipe data yang akan di-sort.
* (5 poin) *function template print()* di mana type parameter T nantinya akan diganti dengan tipe data yang akan dicetak. .
* (5 poin) *main()* program sebagai *drive*r untuk men-tes function templates yang Anda desain menggunakan dua kasus: *array of int* dan *array of double*. Cetak data sebelum di-sort dan sesudah di-sort.

**Jawab:**

1. *class String{*

*private:*

*char \* s;*

*unsigned int len;*

*public:*

*String(unsigned=0);*

*String(const String&);*

*~String();*

*};*

1. *String::String(unsigned=0){strcpy(s, NULL);}*

*String::String(const String& ss){*

*for(int i=0; i<strlen(ss.s); i++) this->s[i]=ss.s[i];*

*this->s[i]='\0';*

*len=strlen(ss);*

*}*

*String::~String(){cout<<”String deleted.”<<endl;}*

1. *template <class T>*

*void sortBubble(T \*items,int count){  
   T t;  
  
 for(int a=1; a<count; a++)  
  for(int b=count-1; b>=a; b--)  
  if(items[b-1] > items[b]) {  
         t = items[b-1];  
         items[b-1] = items[b];  
         items[b] = t;  
       }  
}*

*template <class T>*

*void print(T \*items,int count){  
   T t;*

*for(int i=0; i<count; i++) cout << iarray[i] << endl;  
   cout << endl;*  
 *}*

*int main(){  
  int integers[10] = {3,5,8,12,1,6,2,9,10};  
  double doubles[5] = {0.12,3.5,1.32,7.3,2.22};  
  
  cout<<”Before: “;  
 print(intergers, 10);  
 sortBubble(intergers, 7);  
  
  cout << "After: ";  
 print(intergers,10);  
  cout<<”Before: “;  
 print(doubles, 10);  
 sortBubble(doubles, 7);  
  
  cout << "After: ";  
 print(doubles,10);  
  return 0;  
}*

-ooo-