**Kuis dan TM 7**

**Soal Pilihan**

1. *Inheritance* kerap kali dihubungkan dengan *polymorphism*. Istilah yang paling

 tepat untuk menyatakan inheritance sebagai polymorphism adalah

a.trivial polymorphism b. universal polymorphism

c. pure polymorphism **d. subtype polymorphism**

 02. 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

 03. Salah satu kerugian penggunaan *inheritance* di OOP adalah

 a. independensi di antara class dalam hirarki

 b. independensi di antara object dalam hirarki

 **c. dependensi di antara class dalam hirarki**

 d. dependensi di antara object dalam hirarki

04. *Tipe inheritance* yang dianjurkan untuk melakukan *derivation*  adalah

 a. private b. protected

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

 05. ADT dan *inheritance* mempunyai komponen yang sama tetapi hanya beda strukturnya.

 Paralel di ADT, hirarki di *inheritance*. Karena itu sejumlah inheritance mempunyai

 kelebihan dibandingkan dengan ADT. Kelebihan-kelebihan itu diantaranya adalah

 adalah adanya fitur-fitur bedrikut ini

 a. *reuse-ability* dan *extend ability* b. *extend ability* dan *overriding function*

 c. *overriding function* dan *reuse-ability*

 **d. *reuseability*, *extend ability* dan *overriding function***

**Soal Uraian**

1. Apa yang *salah* dalam penggalan definisi *class* berikut:

*class Time {*

 *public: //* prototype fungsi

 *private: int h = 0;*

 *int m = 0;*

 *int s = 0;*

*}*

 **Jawab:**

*Class tidak boleh menginisialisasi data membernya di luar function member.*

1. Jika *Rasional* adalah *class* bilangan rasional (pecahan seperti 22/7), apa beda ketiga pernyataan berikut:

*Rasional y(x);*

*Rasional y = x;*

*Rasional y; y = x;*

 **Jawab:**

 *Rasional y(x) dan Rasional y=x memanggil copy constructor karena belum diinisialisasi sebelumnya, sedangkan Rasional y; y = x; memanggil assigment operator karena y telah diinisialisasi sebelumnya.*

1. 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.*

1. 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.*

1. Ilmu pengetahuan itu sifatnya kumulatif sehingga meskipun kita sudah mempunyai utilitas ADT yang tidak lain adalah enkapsulasi kita masih memerlukan *inheritance*. WHY? Untuk bisa menjawab kasus ini kamu harus membandingkan *inheritance* dengan ADT. Pertanyaan buat kamu adalah coba bandingkan *inheritance* dengan ADT dari berbagai sisi diantaranya dari sisi *komponen*, *struktur komponen*, *advantage* dan *disadvantage.*

 **Jawab:**

|  |  |  |
| --- | --- | --- |
|  | Inheritance | ADT |
| Komponen | Constructor, copy constructor, destructor, data member and function, access modifier | Data member and function, access modifier |
| Struktur komponen | Turunan | Paralel |
| Advantage | Reusability, overriding, extendability | Independen  |
| Disadvantage | Dependen | Independen  |

 **I AM PROUD AS BINUSIAN**