Programmazione ad oggetti – Laurea in Informatica
Appello d'Esame – 4/12/2009

Nome..... Cognome.....

Matricola..... Laurea in.....

Non si possono consultare appunti e libri. Dove previsto scrivere CHIARAMENTE la risposta nell'apposito spazio.

Quesito 1

Si consideri il seguente modello di realtà. Lo smartphone di $e^{i\pi}$-esima generazione aiPhone$^{\copyright}$ permette di installare un insieme di applicazioni gratuite oppure a pagamento.

1. Definire la seguente gerarchia di classi.
   (a) Definire una classe `App` i cui oggetti rappresentano una applicazione installabile su un aiPhone$^{\copyright}$. Ogni `App` è caratterizzata dalla memoria richiesta (in MB) per l'installazione e dalla necessità o meno dell'applicazione di accedere alla rete Internet. Dotare la classe `App` di opportuno/i costruttore/i.
   (b) Definire una classe `FreeApp` derivata da `App` i cui oggetti rappresentano una applicazione gratuita installabile su un aiPhone$^{\copyright}$. Ogni `FreeApp` è caratterizzata dall'essere oppure no una applicazione open source. Dotare la classe `FreeApp` di opportuno/i costruttore/i.
   (c) Definire una classe `PayApp` derivata da `App` i cui oggetti rappresentano una applicazione a pagamento installabile su un aiPhone$^{\copyright}$. Ogni `PayApp` è caratterizzata dal prezzo di acquisto (in €). Dotare la classe `PayApp` di un costruttore che includa un parametro formale `p` di tipo `double` per costruire una `PayApp` dal prezzo di acquisto `p`: nel caso in cui il parametro attuale per `p` sia ≤ 0, tale costruttore deve sollevare una eccezione di tipo `Exc` (una classe di eccezioni di cui è richiesta la definizione).
2. Definire una classe `aiPhone` i cui oggetti rappresentano un aiPhone$^{\copyright}$. Più precisamente, un oggetto `aiPhone` è caratterizzato dalle applicazioni installate, che sono rappresentate mediante un contenitore di puntatori al tipo `App`, e dalla capacità massima di memoria (in MB) dell'aiPhone$^{\copyright}$. Devono essere disponibili le seguenti funzionalità:
   (a) un metodo `double installaApp(const App&)` con il seguente comportamento: una invocazione `ai.installaApp(a)` installa l'applicazione `a` qualora la memoria residua dell'aiPhone$^{\copyright}$ `ai` lo consenta e quindi ritorna la memoria residua di `ai` dopo tale installazione; altrimenti, l'applicazione `a` non viene installata in `ai` e viene invece sollevata una eccezione di tipo `Exc`.
   (b) un metodo `vector<PayApp> F(double)` con il seguente comportamento: una invocazione `ai.F(x)` ritorna un vector (eventualmente vuoto) contenente una copia di tutte le applicazioni a pagamento installate nell'aiPhone$^{\copyright}$ `ai` che (1) accedono alla rete Internet e (2) hanno un prezzo di acquisto $\leq$ `x`. Nella definizione di tale metodo **non è possibile** usare l'operatore di indicizzazione su qualsiasi contenitore.
   (c) un metodo `FreeApp* G()` con il seguente comportamento: una invocazione `ai.G()` ritorna un puntatore sempre non nullo ad una applicazione gratuita installata nell'aiPhone$^{\copyright}$ `ai` che (1) sia open source e (2) richieda la massima memoria per l'installazione tra tutte le applicazioni open source installate nell'aiPhone$^{\copyright}$ `ai`; se una tale applicazione gratuita non dovesse esserci tra le applicazioni installate nell'aiPhone$^{\copyright}$ `ai` allora viene sollevata una eccezione di tipo `Exc`.

NB: Scrivere la soluzione **chiaramente** nel foglio a quadretti. Per comodità di correzione, definire tutti i metodi inline.

Quesito 2

Scrivere un programma esempio che dimostri con opportune stampe su `cout` la presenza del virtual pointer in una classe polimorfa.

Quesito 3

Scrivere un programma consistente di esattamente tre classi `A`, `B` e `C` e della sola funzione `main()` che soddisfi le seguenti condizioni:

1. la classe `A` è definita come:

```
class A { public: virtual ~A(){} };
```

2. le classi `B` e `C` devono essere definite per ereditarietà e non contengono alcun membro
3. la funzione `main()` definisce le tre variabili:

   `A* pa = new A; B* pb = new B; C* pc = new C;`

   e nessuna altra variabile (di alcun tipo)

4. la funzione `main()` può **utilizzare solamente** espressioni di tipo `A*`, `B*` e `C*`, **non può** sollevare eccezioni mediante una `throw` e **non può** invocare l'operatore `new`
5. il programma deve compilare correttamente
6. l'esecuzione di `main()` **deve provocare un errore run-time**.

Quesito 4

Si scriva una gerarchia di almeno 5 classi che includa una classe `D` definita mediante ereditarietà a diamente ed una classe `B` che sia una classe base virtuale per `D`. Scrivere inoltre un programma esempio che dimostri con opportune stampe su `cout` che i costruttori delle superclassi dirette non virtuali di `D` escludono di richiamare il costruttore della classe base virtuale `B`.