UNIVERSITÀ DEGLI STUDI DI PISA
Corso di Laurea in Ingegneria Informatica
c/o Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni

Esame di Ingegneria dei Sistemi Software

Appello del 15 gennaio 2007

Nome e cognome:
Matricola:

Il punteggio in trentesimi associato a ciascuna domanda è indicato fra parentesi.

Scegliere una risposta per ciascuna domanda:

1	**Una classe si dice *concreta* se**	(1)
	descrive oggetti concreti.	☐
	è interamente implementata.	☒
	non ha classi derivate.	☐
2	**Con l'eredità multipla**	(1)
	una classe può derivare direttamente da piú basi.	☒
	una classe può avere piú classi derivate.	☐
	una classe ha piú di una implementazione.	☐
3	**Il controllo di qualità avviene**	(1)
	nella fase iniziale del processo di sviluppo,	☐
	nel corso di tutto il processo di sviluppo.	☒
	nella fase di convalida e verifica.	☐
4	**L'insieme delle formule di un linguaggio è definito**	(1)
	dalla sintassi.	☒
	dalla semantica.	☐
	dalle regole d'inferenza.	☐
5	**In UML l'architettura fisica si rappresenta**	(1)
	con i diagrammi delle classi.	☐
	con i diagrammi di collaborazione.	☐
	con i diagrammi di deployment.	☒

Rispondere alle domande, usando solo lo spazio disponibile:

6 Che cosa significa *visibilità protetta*? (3)

Un'entità con visibilità protetta è accessibile solo dalle classi derivate.

7 A cosa servono i moduli generici? (3)

A rappresentare la struttura comune di algoritmi e classi, mettendone in evidenza le parti variabili per mezzo di parametri.

8 A cosa serve il Servizio Eventi nelle architetture CORBA? (3)

Ad ottenere un accoppiamento meno stretto fra componenti distribuiti, permettendo lo scambio di eventi asincroni.

9 Scrivere la tabella di verità della formula proposizionale $(A \wedge B) \Rightarrow B$. È valida? (3)

A	B	$A \wedge B$	$(A \wedge B) \Rightarrow B$
T	**T**	T	$T \Rightarrow T = T$
T	**F**	F	$F \Rightarrow F = T$
F	**T**	F	$F \Rightarrow T = T$
F	**F**	F	$F \Rightarrow F = T$

È valida.

10 Negli Statechart, che cos'è una guardia? (3)

Una condizione che deve essere vera affinché una transizione possa avvenire.

11 Disegnare un diagramma UML corrispondente al codice di Fig. 1. (4)

12 Con riferimento all'esercizio precedente: (6)

I parametri di `Adapter::upcall()` sono l'identificatore di un oggetto di tipo derivato da `SvtBase` e il nome di un'operazione che deve essere eseguita da tale oggetto.
L'operazione `Adapter::add_svt()` aggiunge un oggetto di tipo derivato da `SvtBase` e ne restituisce l'identificatore.
L'operazione `Greeting_skl::invoke()` riceve in ingresso il nome di un'operazione ed esegue l'operazione corrispondente, implementata da una classe derivata da `IGreeting`.
Implementare l'operazione `Greeting_skl::invoke()`.
Scrivere un programma principale che stampi i messaggi "Ciao!" e "Hello!" usando `Adapter::upcall()`.

```
class SvtBase {
public:
    virtual void invoke(string n) = 0;
};
class Adapter {
    vector<SvtBase*> svt_;
public:
    int add_svt(SvtBase* s);
    void upcall(int oid, string n);
};
class IGreeting {
public:
    virtual void hello() = 0;
    virtual void ciao() = 0;
};
class Greeting_skl : public IGreeting,
                     public SvtBase {
public:
    virtual void invoke(string n);
};
class Greeting_svt : public Greeting_skl {
    virtual void hello();
    virtual void ciao();
};
```

```
void Adapter::
upcall(int s, string n)
{
    svt_[s]->invoke(n);
}
int Adapter::
add_svt(SvtBase* s)
{
    svt_.push_back(s);
    return (svt_.size() - 1);
}
void Greeting_svt::
hello()
{
    cout << "Hello!" << endl;
}
void Greeting_svt::
ciao()
{
    cout << "Ciao!" << endl;
}
```

Figura 1: Domanda 11.

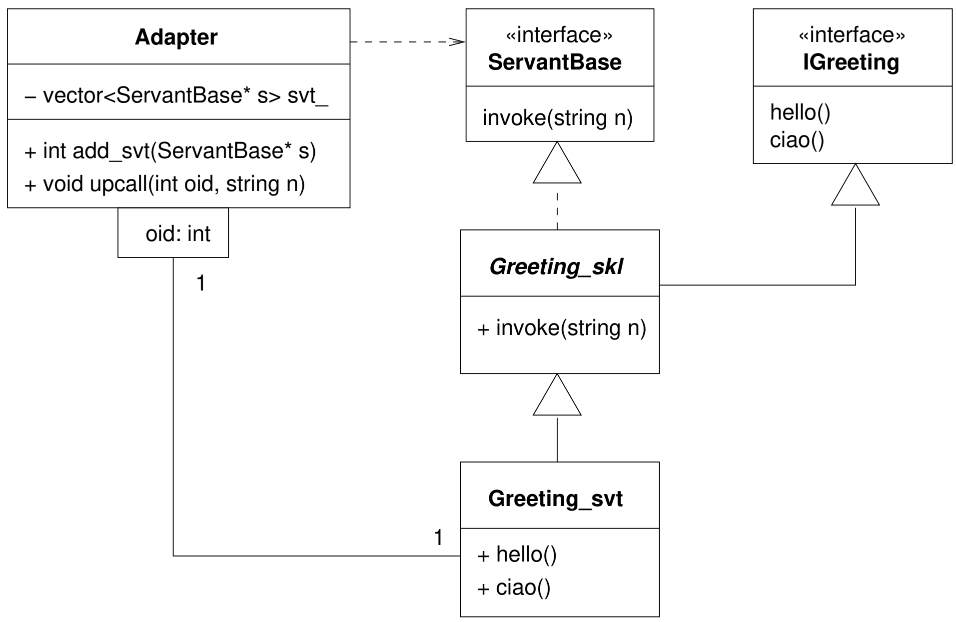

Figura 2: Domanda 11, soluzione.

```
void
Greeting_skl::
invoke(string n)
{
    if (n == "hello") {
        this->hello();
    } else if (n == "ciao") {
        this->ciao();
    }
}

int
main()
{
    Adapter a;
    Greeting_svt g;
    int svt = a.add_svt(&g);
    a.upcall(svt, "hello");
    a.upcall(svt, "ciao");
}
```

Figura 3: Domanda 12, soluzione.