UNIVERSITÀ DEGLI STUDI DI PISA

Corso di Laurea in Ingegneria Informatica

c/o Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni

Esami di Ingegneria del Software
e di Ingegneria dei Sistemi Software

Appello del 20 luglio 2007

Nome e cognome:
Matricola:
Esame: **Ing. del Software** ☐ **Ing. dei Sistemi Software** ☐

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

Scegliere una risposta per ciascuna domanda:

1	**Quale di queste ER riconosce gli identificatori del C++?**	(1)
	`[_A-Za-z][_A-Za-z0-9]*`	☒
	`[_A-Za-z]([_A-Za-z][0-9])*`	☐
	`[_A-Za-z][_A-Za-z0-9]+`	☐
2	**In fase di analisi dei requisiti si possono usare**	(1)
	soltanto i diagrammi delle classi.	☐
	soltanto i diagrammi dei casi d'uso.	☐
	i diagrammi delle classi e dei casi d'uso.	☒
3	**In un sistema formale completo**	(1)
	tutte le formule dimostrabili sono valide.	☐
	tutte le formule valide sono dimostrabili.	☒
	tutti gli assiomi sono validi.	☐
4	**Nascondere l'implementazione di un modulo**	(1)
	serve a proteggere la proprietà intellettuale.	☐
	serve a ridurre le dipendenze fra moduli.	☒
	serve a impedire un uso scorretto del modulo.	☐
5	**Un sistema si dice *robusto* se**	(1)
	soddisfa i requisiti.	☐
	si comporta accettabilmente in condizioni non previste.	☒
	sopporta i sovraccarichi.	☐

Rispondere alle domande, usando solo lo spazio disponibile:

6 Che cosa sono i modelli di processo evolutivi? (3)

Modelli di processi in cui il software viene sviluppato incrementalmente in passi successivi, ognuno dei quali produce una parte o una versione del sistema.

7 Che cos'è la manutenzione del software? (3)

Riparazione dei difetti nel prodotto consegnato al cliente, o aggiornamento del codice per fornire nuove versioni, attraverso la modifica e la ricostruzione del software.

8 Quali sono i meccanismi di estensione del linguaggio UML? (3)
Vincoli, stereotipi, e valori etichettati (o proprietà).

9 Che cos'è un'architettura software? (3)

Una descrizione della struttura del sistema da realizzare, espressa in termini dei suoi componenti e delle loro relazioni reciproche.

10 Che cosa significa la formula $M\,[t\rangle \wedge M\,[u\rangle \wedge \exists_{p \in {}^\bullet t \cap {}^\bullet u}(M(p) < W(\langle p,t\rangle) + W(\langle p,u\rangle))$ **?** (3)

Le transizioni t ed u sono abilitate nella marcatura M ed esiste un place p di ingresso a t ed u tale che la sua marcatura sia minore della somma dei pesi degli archi che lo uniscono a t ed u. Le due transizioni sono quindi in *conflitto effettivo*.

11 Disegnare un diagramma di classi che descriva quanto segue: (5)
Un'*espressione* può essere valutata, ottenendo un valore intero. Un'espressione può essere un *letterale*, una *variabile*, o un'*espressione binaria*. Un letterale ha una *rappresentazione* (stringa di caratteri) e un *valore*, una variabile ha un *nome* (stringa di caratteri) e un valore; un'espressione binaria è formata da due espressioni ed ha un *operatore* (simbolo di operazione aritmentica, rappresentato da un tipo **ArithOp** non ulteriormente specificato).

12 Implementare in C++ le classi definite nell'esercizio precedente. (5)
(Usare la classe predefinita `string` per le stringhe di caratteri e usare un'enumerazione per il tipo **ArithOp**. Usare i costruttori per inizializzare le istanze delle varie classi).

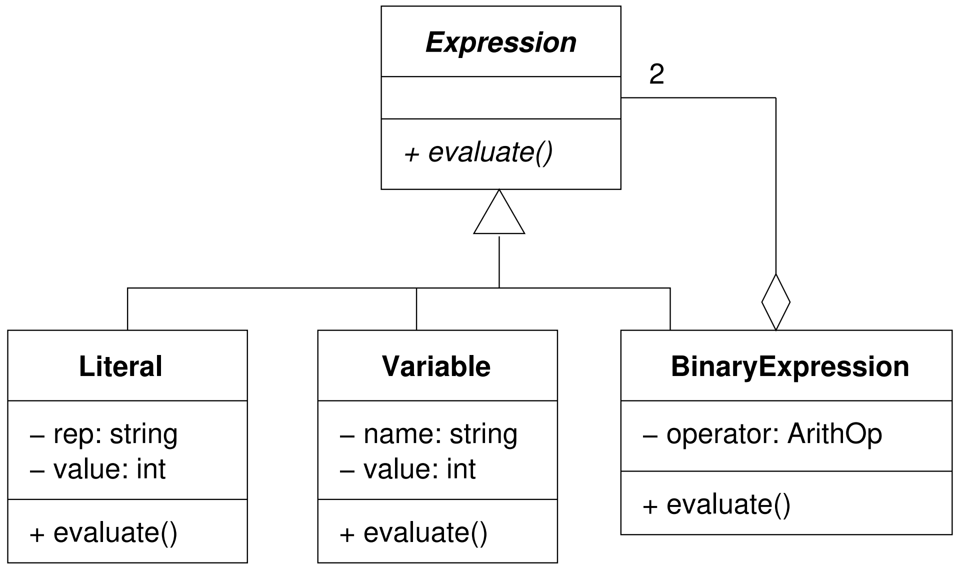

Figura 1: Domanda 11.

```
#include <string>
using namespace std;

enum ArithOp { PLUS, MINUS, TIMES, DIV, REM, };

class Expression {
public:
    virtual int evaluate() = 0;
};

class Literal : public Expression {
    string rep_;
    int value_;
public:
    Literal(string r, int v);
    int evaluate();
};

class Variable : public Expression {
    string name_;
    int value_;
public:
    Variable(string n, int v);
    int evaluate();
};

class BinaryExpression : public Expression {
    ArithOp operator_;
    Expression* left_;
    Expression* right_;
public:
    BinaryExpression(ArithOp op, Expression* l, Expression* r);
    int evaluate();
};
```

Figura 2: Domanda 12 (dichiarazioni).

```
#include <iostream>
#include <string>
using namespace std;

Literal::
Literal(string r, int v) : rep_(r), value_(v) {};

int
Literal::
evaluate()
{
    return value_;
}

Variable::
Variable(string n, int v) : name_(n), value_(v) {};

int
Variable::
evaluate()
{
    return value_;
}

BinaryExpression::
BinaryExpression(ArithOp op, Expression* l, Expression* r)
    : operator_(op), left_(l), right_(r) {};

int
BinaryExpression::
evaluate()
{
    switch (operator_) {
    case PLUS: return (left_->evaluate() + right_->evaluate());
    case MINUS: return (left_->evaluate() - right_->evaluate());
    case TIMES: return (left_->evaluate() * right_->evaluate());
    case DIV: return (left_->evaluate() / right_->evaluate());
    case REM: return (left_->evaluate() % right_->evaluate());
    }
}
```

Figura 3: Domanda 12 (definizioni).