Esame di Ingegneria del software

Appello del 16 febbraio 2018

Nome e cognome:
Matricola:

Il punteggio relativo a ciascuna domanda, indicato fra parentesi, è in trentesimi. I candidati devono consegnare entro un'ora dall'inizio della prova.

1 **Disegnare il diagramma di classi** relativo al seguente problema: la classe astratta **Network** è un aggregato di istanze della classe **Node**, con le operazioni *addNode()* e *removeNode()* per aggiungere e togliere nodi, e un'operazione *createIterator()* che restituisce un'istanza di una classe derivata dalla classe astratta **Iterator**. Quest'ultima permette di accedere in sequenza ai nodi appartenenti ad un oggetto **Network**, per mezzo delle operazioni *first()* e *next()*. L'operazione *isDone()* serve a sapere se l'enumerazione dei nodi è terminata. La classe **Network** viene realizzata dalle classi **Vector** e **Database**, che definiscono contenitori concreti per mezzo di diverse strutture dati. I loro metodi createIterator() creano e restituiscono istanze, rispettivamente, delle classi concrete **VectorIter** e **DBIter**. (5)

2 **Scrivere un programma in C++** che, usando il framework di Fig. 1, (i) crei una finestra col titolo "Main Window"; (ii) vi inserisca un bottone e un campo di testo; (iii) scriva sull'uscita standard il contenuto del campo testo, quando viene premuto il bottone. (5)

3 **Disegnare uno statechart UML** che specifichi quanto segue. Un file può essere di tipo volatile o persistente; un file persistente può essere locale o su nastro, mentre un file volatile non viene mai messo su nastro; un file è locale mentre viene creato o copiato su o da memoria RAM, e mentre è memorizzato su disco; al termine della creazione o copia, il file va su nastro se permanente o su disco se volatile; il comando read copia un file da disco a memoria, il comando load sposta un file da nastro a disco, il comando store sposta da disco a nastro, il comando remove sposta un file permanente da disco a nastro, o cancella un file volatile; i file su nastro devono essere trasferiti su disco per essere letti. (5)

4 **Con riferimento alla Fig. 2, rispondere alle domande**. (5)

	V	**F**
Statement implementa **Codesegment**.	☐	☒
una **Expression** può contenere dei **CodeSegment**.	☐	☒
una **Instruction** fa parte di un **CodeSegment**.	☒	☐
tutti gli *Statement* sono **Expression**.	☐	☒
generate() restituisce un oggetto di tipo **CodeSegment**.	☒	☐

5 Disegnare un diagramma di classi relativo al seguente problema: Si hanno delle classi **Square**, **Circle** etc., che rappresentano figure geometriche. Ognuna di esse ha un'operazione area():int che restituisce l'area della figura. (a) Si definisca l'interfaccia di una classe **Container** che contenga un insieme di figure qualsiasi e che permetta di i) aggiungere una figura, ii) ottenere un riferimento (o un puntatore) a una figura individuata da un indice, e iii) ottenere il numero di figure contenute; (b) si definisca una classe **Client** con un'operazione total_area che restituisce l'area totale, mostrandone l'implementazione. (5)

6 **Disegnare un diagramma di stato** che modelli il comportamento della seguente classe C++: (5)

```
class Buffer2 {
    enum state { EMPTY, HALF, FULL };
    state s;
public:
    Buffer2() : s(EMPTY) {};
    void data_in();
    void data_out();
};

void
Buffer2::
data_in()
{
    switch (s) {
    case EMPTY: s = HALF; break;
    case HALF:  s = FULL; break;
    case FULL:  break;
    }
};

void
Buffer2::
data_out()
{
    switch (s) {
    case EMPTY: break;
    case HALF:  s = EMPTY; break;
    case FULL:  s = HALF; break;
    }
};
```

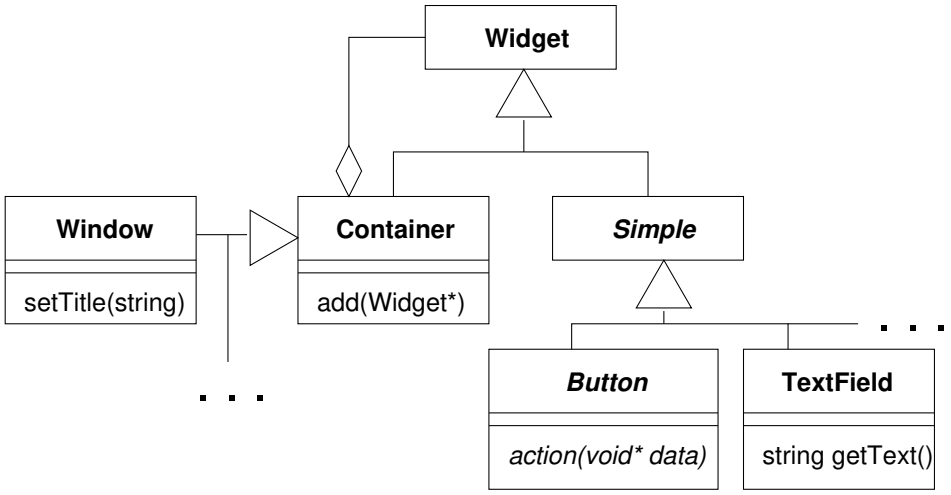

Figura 1: Domanda 2.

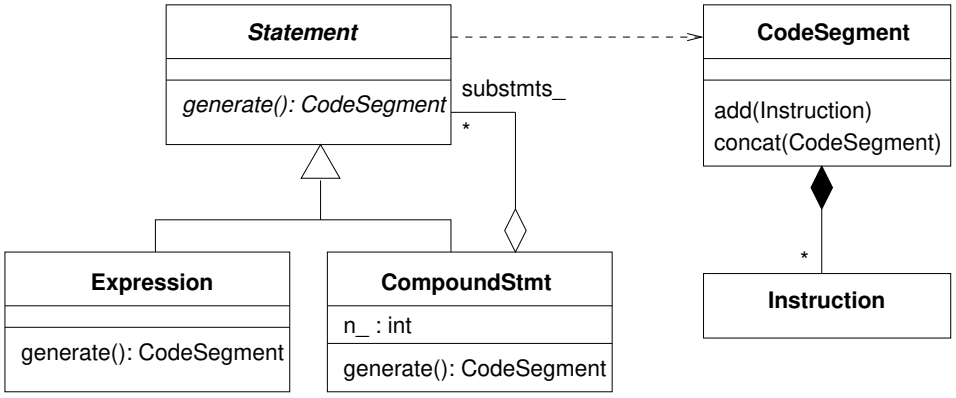

Figura 2: Domanda 4.

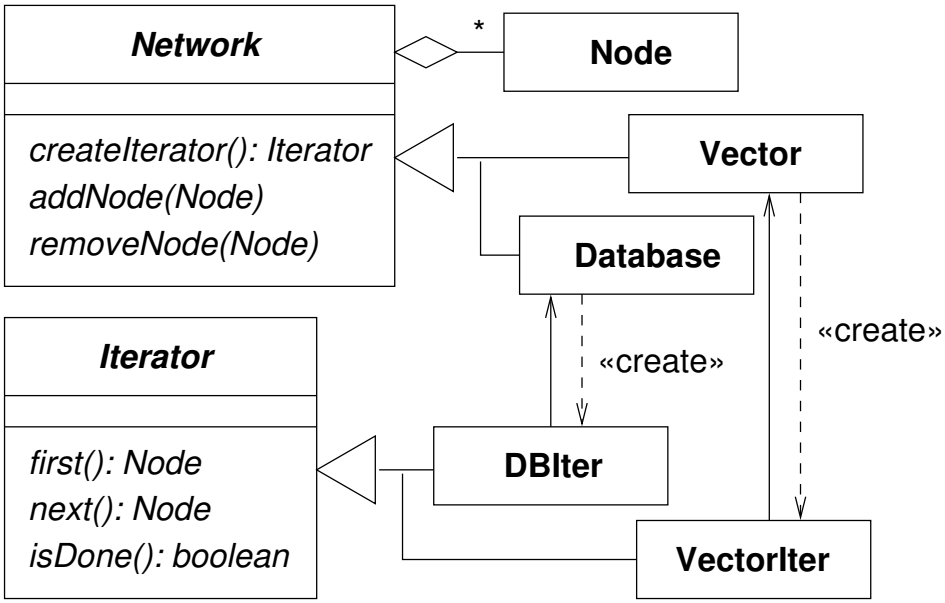

Figura 3: Domanda 1, soluzione.

```
#include <iostream>
#include "widgets.h"

using namespace std;

class MyButton : public Button {
    TextField* tf;
public:
    MyButton(TextField* t) : tf(t) {};
    void action(void* data);
};

void
MyButton::
action(void* data)
{
    cout << tf->getText() << endl;
}

int
main()
{
    Window* w = new Window;
    TextField* t = new TextField;
    MyButton* b = new MyButton;

    w->setTitle("Main Window");
    w->add(t);
    w->add(b);
}
```

Figura 4: Domanda 2, soluzione.

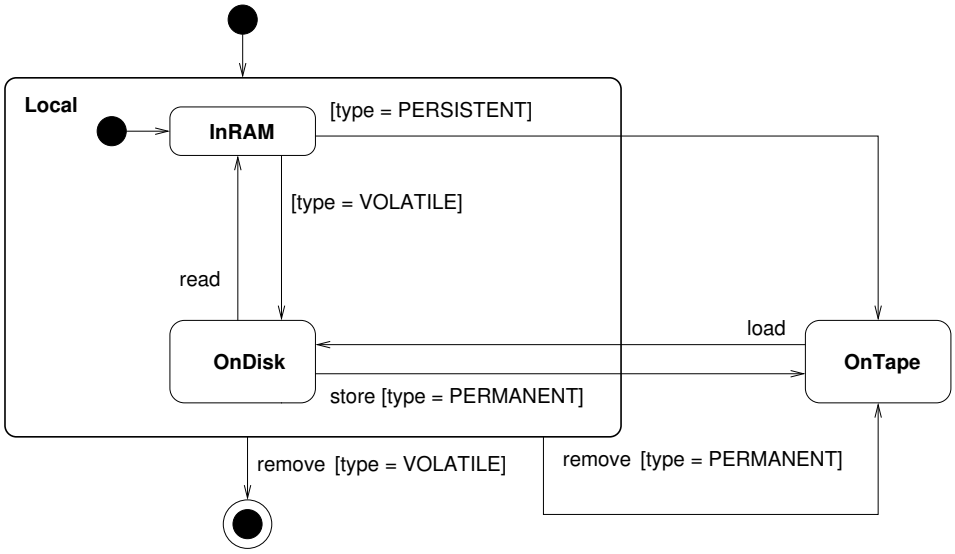

Figura 5: Domanda 3, soluzione.

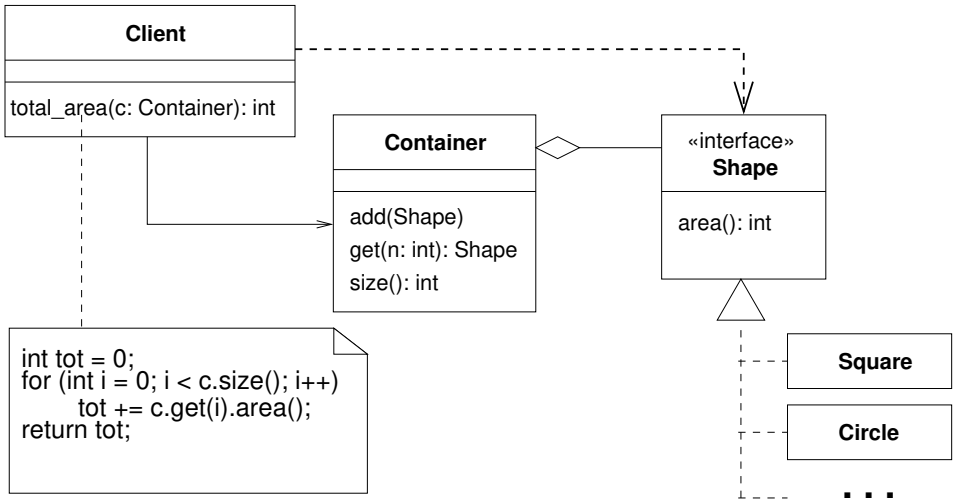

Figura 6: Domanda 5, soluzione.

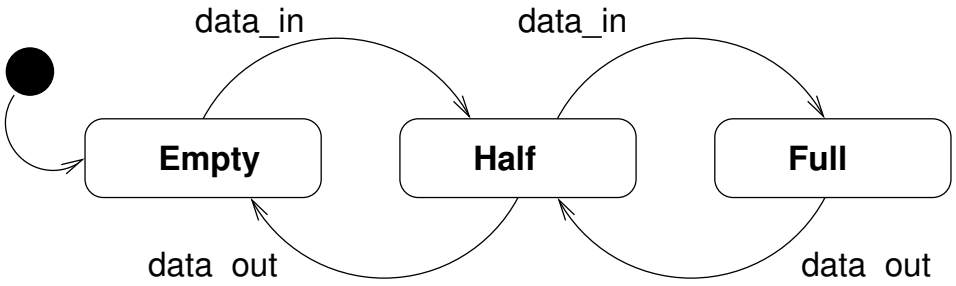

Figura 7: Domanda 6, soluzione.