Programmazione procedurale e ad oggetti - 23/07/2018

Nome: .....
Cognome: .....
Matricola: .....

1. Data la struttura `struct Rational{int num; int denum;}`, scrivete una procedura `Swap` per effettuare lo scambio di due variabili `a` e `b` di tipo `struct Rational`.

Sol. Il C supporta unicamente il passaggio per copia (o valore). Di conseguenza la funzione deve avere come parametri due puntatori alle strutture. In altri termini, vengono passati (per copia) gli indirizzi di `a` e `b`, ovvero la chiamata sarà `Swap(&a,&b)`.

```
void Swap(struct Rational *a,struct Rational *b)
{int num,denum;
 if(a!=null&&b!=null){
 num=a->num;denum=a->denum;
 a->num=b->num;a->denum=b->denum;
 b->num=num;b->denum=denum;}
 return;
}
```

2. Illustrate il layout del record di attivazione di procedura in C.

Sol. Si vedano i lucidi del corso.

3. La *traccia* di una matrice è definita come la somma degli elementi sulla diagonale principale. Scrivete una funzione che accetta in ingresso una matrice `A` (quadrata) di `short`, il suo ordine `n`, e ne restituisce la traccia. Quali condizioni deve soddisfare il parametro attuale associato alla matrice?

```
int Traccia(short** mat,int n)
{int traccia=0;int i;
 if(mat!=null){
  for(i=0;i<n;i++)traccia+=mat[i][i];
  return traccia;}
 else return ERR; //codice d'errore
}
```

La funzione deve essere chiamata con il primo parametro attuale che deve essere un vettore di `n` puntatori a `short`, dove ogni singolo puntatore contiene l'indirizzo base di un vettore di `n` oggetti di tipo `short`.

4. Che differenza c'è tra binding dinamico e statico? Il binding dinamico è supportato in C++? Se sì, in che modo?

Sol. Per binding si intende il processo di associazione nome-di-procedura indirizzo. Il binding è detto dinamico se tale associazione viene fatta durante l'esecuzione (in contrapposizione al binding statico - in compilazione), ovvero solo al momento dell'esecuzione è noto l'indirizzo della procedura da eseguire a fronte di una data chiamata. Il C++ supporta il binding dinamico attraverso il meccanismo delle funzioni virtuali. Per ogni funzione dichiarata `virtual` viene riservato un puntatore destinato a contenere l'indirizzo preciso della funzione da eseguire a fronte della chiamata. Detto in altri termini, a fronte della chiamata `obj.f()` il compilatore genera del codice che prevede un salto all'istruzione il cui indirizzo è contenuto nel puntatore associato alla funzione virtuale `f`.

5. Mostrate tramite esempi i possibili usi della parola riservata `virtual`.

Sol. Si vedano gli esempi sui lucidi (e sui sorgenti mostrati a lezione e disponibili su moodle) relativi alle funzioni virtuali e alle classi base virtuali.

6. Considerate il seguente programma e determinatene l'output (sequenza di messaggi stampati in esecuzione). In generale, quali sono le regole che determinano l'ordine secondo il quale costruttori e distruttori sono chiamati?

```
class A {public: A(int a){std::cout<<"Costruttore chiamato per:"<<a<<'\n';el=a;};
         ~A(){std::cout<<"Distruttore chiamato per:"<<el<<'\n';};int el;};

class B:public A{public B(int b):A(b-1){std::cout<<"Costruttore chiamato per:"<<b<<'\n';el=b;};
                 ~B(){std::cout<<"Distruttore chiamato per:"<<el<<'\n';};
                 int el; A objA=A(5);};
A objA=A(8);
int main(void){B objB=B(2);return 0;}
```

Sol. Il programma stampa:

```
Costruttore chiamato per:8
Costruttore chiamato per:1
Costruttore chiamato per:5
Costruttore chiamato per:2
Distruttore chiamato per:2
Distruttore chiamato per:5
Distruttore chiamato per:1
Distruttore chiamato per:8
```

In generale, al momento della creazione di un oggetto di una classe derivata (`objB` nell'esercizio) viene prima chiamato il costruttore della classe base (`A` nell'esercizio), poi quelli per gli eventuali oggetti presenti come attributi nella classe (`objA` nell'esercizio), e solo in ultimo il costruttore della classe derivata. Nel momento in cui un oggetto esce dallo scope (locale o globale che sia) si osserva il processo inverso, ovvero una sequenza di chiamate ai costruttori nell'ordine inverso rispetto a quello osservato per i costruttori.