Esercizi con esempi di svolgimento

Istruzioni per l'uso e avvertenze

Per utilizzare al meglio i seguenti esercizi, per ognuno di essi si esegua il seguente algoritmo:

1. Leggere bene la traccia
2. Rileggere bene la traccia
3. Cercare di capire chiaramente quanto richiesto. Se perplessi tornare al punto 1.
4. Progettare il programma. Una possibile sequenza è:
   (a) dichiarare le strutture necessarie
   (b) individuare le funzioni necessarie e dichiararne i prototipi
   (c) scrivere il main
   (d) scrivere le funzioni individuate al punto precedente
5. Compilare e leggere con attenzione eventuali messaggi del compilatore
6. Correggere eventali errori
7. Controllare la correttezza del programma, eventualmente tornando al punto 4. o al punto 1.
8. SOLO una volta che il programma realizzato si ritiene corretto o se, viceversa, si è superato il limite di 1000 iterazioni nei cicli precedenti, consultare la soluzione proposta (che, si ricorda, una POSSIBILE soluzione, non LA soluzione; è quindi perfettamente inutile impararla a memoria)

Ogni soluzione è fornita come un file `.zip` contenente la directory all'interno della quale è il project NetBeans. Il file va quindi espanso salvando la directory in essa contenuta nella directory `NetBeansProjects` presente nella raccolta Documenti. Al successivo avvio di NetBeans, il progetto dovrebbe essere accessibile con il comando `Open Project` (o simile in italiano) sotto il menù `File`.

Esercizio A

Si scriva un programma che definisca e utilizzi la struttura dati più adatta per gestire un elenco di corsi universitari, in cui il generico elemento deve contenere i seguenti dati:

- nome del corso
- nome del docente
- numero di crediti
- numero di studenti iscritti al corso

Il programma deve prevedere inizialmente la lettura dell'elenco con tutti i dati per un numero N di corsi specificato dall'utente. Successivamente, il programma deve ordinare l'elenco in senso crescente per numero di studenti iscritti e poi prevedere la lettura di un valore intero K. Il programma deve quindi realizzare K letture dei dati di ulteriori corsi che vanno inseriti, uno alla volta, nell'elenco ordinato. Al termine, si stampi l'elenco cos modificato, riportando tutti i dati di ciascun elemento.
Costituisce elemento di valutazione positiva la progettazione modulare del programma, l'utilizzo di tecniche per l'allocazione dinamica della memoria, il commento opportunamente dettagliato del codice, la realizzazione di I/O adeguato.
Commento: La soluzione proposta è disponibile nel file `esempio1.zip` e gestisce gli array non utilizzando allocazione di memoria dinamica e puntatori. Il programma proposto è organizzato in moduli separati.

Esercizio B

Si scriva un programma che definisca e utilizzi la struttura dati più adatta per gestire un elenco di aziende, in cui il generico elemento deve contenere le seguenti informazioni:

- nome dell'azienda
- tipologia ("meccanica", "tessile", "elettronica", "farmaceutica")
- numero di dipendenti
- capitale sociale (in migliaia di euro, valore reale)

Il programma deve prevedere inizialmente la lettura di un elenco con tutti i dati per un numero N di aziende specificato dall'utente. Successivamente si preveda l'input da parte dell'utente di una tipologia (tra quelle elencate sopra) e di un valore reale T. Il programma individui quindi tutti gli elementi corrispondenti alla tipologia inserita ed avente capitale sociale maggiore di T, trasferendoli in un nuovo elenco; gli altri elementi della tipologia scelta ma con capitale sociale minore o uguale di T siano invece eliminati.
Il programma infine disponga in ordine alfabetico i due elenchi secondo il nome dell'azienda e stampi i due elenchi cos ottenuti, fornendo in output, per ciascuna azienda, tutti i suoi dati.
Costituisce elemento di valutazione positiva la progettazione modulare del programma, l'utilizzo di tecniche per l'allocazione dinamica dela memoria, il commento adeguato del codice, la realizzazione di I/O adeguato.
Commento: La soluzione proposta è disponibile nel file `esempio2.zip` e gestisce gli array non utilizzando allocazione di memoria dinamica e puntatori. Il programma proposto è organizzato in moduli separati.

Esercizio C

Si scriva un programma che definisca e utilizzi la struttura dati più adatta per gestire un elenco di prenotazioni gestito da una compagnia aerea in cui il generico elemento deve contenere le seguenti informazioni:

- identificativo del volo (es. AZ295)
- cognome del cliente
- nome del cliente
- codice di prenotazione (valore di tipo intero)

Il programma deve prevedere inizialmente la lettura dell'elenco con tutti i dati per un numero N di clienti specificato dall'utente. Al termine della lettura, il programma deve prevedere la lettura di due valori K1 e K2 sempre specificati dall'utente e formare due nuovi elenchi: il primo contenente le prime K1 prenotazioni in ordine di codice ed il secondo contenente le ultime K2. Il programma deve infine stampare i due elenchi ottenuti, fornendo in output le iniziali del cliente, l'identificativo del volo e il codice di prenotazione.
Costituisce elemento di valutazione positiva la progettazione modulare del programma, l'utilizzo di tecniche per l'allocazione dinamica dela memoria, il commento opportunamente dettagliato del codice, la realizzazione di I/O adeguato.
Commento: La soluzione proposta è disponibile nel file `esempio3.zip` ed stata realizzata da uno studente come compito di esame. È piuttosto sofisticata, con ampio uso di memoria dinamica e puntatori. Vale la pena studiarsela con attenzione, soprattutto per quanto riguarda l'organizzazione in moduli separati.