Manuale per il docente:

Linee guida attività “Allocazione dinamica della memoria”:

1. Introduzione:

Le strutture dati nel linguaggio C, cosi come in altri linguaggi, sono normalmente di dimensione fissa, e sono quelle che chiamiamo strutture dati "statiche". Per esempio, il numero di elementi di cui è costituito un vettore, è fissato una volta che il programma è stato compilato. Tuttavia, questo aspetto può risultare negativo da un lato: se vogliamo modificare il numero degli elementi che vanno a comporre, ad esempio, un array, dobbiamo per forza di cose andare a modificare tutto il programma. In questa attività, quindi, viene presentato un potente strumento per far fronte a quanto descritto: L'Allocazione dinamica della memoria. Essa si basa, sostanzialmente, su 3 istruzioni principali:

1- malloc: Tale funzione è in grado di allocare un blocco di memoria su cui punta una opportuna variabile puntata, senza inizializzarlo.

2- calloc: Tale funzione è in grado di allocare un blocco di memoria su cui punta una opportuna variabile puntata, inizializzandolo a 0.

3- realloc: Tale funzione è in grado di ridimensionare un blocco di memoria allocato precedentemente su un altro spazio di memoria.

Se si pensa ad un tipo di struttura dati "dinamica", si pensa subito al concetto di "lista", ed in particolare al concetto di lista concatenata (LinkedList). Una lista concatenata è una struttura dati costituita da oggetti chiamati nodi. Ogni nodo può contenere dati ed un riferimento ad un altro nodo, in modo da formare, appunto, una lista.

Diversi sono invece gli ArrayList. Questi sono, infatti, strutture dinamiche basate su array e quindi diverse dalle liste concatenate. Liste concatenate ed arraylist differiscono, dunque, soprattutto in termini di tempi di accesso e manipolazione dei dati in esse contenuti. L'obiettivo di questa attività è di presentare delle implementazioni di queste strutture dinamiche e le loro manipolazioni, andando a distinguerne gli aspetti fondamentali.

1. Prerequisiti e Modelling:

Una volta introdotti i concetti sopra riportati, un buon punto di partenza è proporre un esercizio in cui vengono messi in ballo le istruzioni “malloc”, “calloc” e “realloc”, spiegando cosa permettono di fare, effettivamente, queste istruzioni. Come esempio è suggerito la classica allocazione dinamica di un array (ad esempio un array di interi), su cui svolgere una qualsivoglia operazione (ad esempio inserire dinamicamente un certo numero di interi e vedere quali di questi sono primi, per poi inserirne dinamicamente altri). Successivamente è opportuno parlare di liste, che rappresentano il più classico esempio di struttura dinamica. A tal proposito, è opportuno parlare di liste concatenate (o LinkedList), ed in particolare della loro rappresentazione ed implementazione (ossia della classica coppia valore-nodo).

È opportuno fare dei riferimenti verso Java in merito alle LinkedList ed ArrayList, anche perché, parallelamente a questo corso, viene svolto il corso avanzato di Java “Object Oriented Software Design”, e proprio concetti come LinkedList ed ArrayList sono al centro della discussione.

1. Attività e Coaching:

Di seguito, ecco un elenco di attività da proporre agli studenti:

1. Primo esempio:

Dati una sequenza di interi forniti da terminale, usare opportunatamente le funzioni "malloc" e "realloc" per allocare dinamicamente un array che andrà a contenere questi elementi, per poi ordinarli in ordine crescente. Deve essere possibile, inoltre, inserire altri elementi oltre quelli inseriti in precedenza.

Gli esempi che seguono, devono essere presentati solo dopo aver introdotto e discusso le liste. Considerando il fatto che, in parallelo a questo corso, viene tenuto anche un corso di Java “avanzato” (Object Oriented Software Design), è interessante vedere le diverse implementazioni che si hanno delle LinkedList e degli ArrayList, ed in particolare nel linguaggio C.

Coaching: In questo caso si deve prestare particolare attenzione all’uso delle due funzioni malloc e realloc (anche in vista degli esercizi successivi). Una difficoltà tipica che si può incontrare in esempi come questo, è quella di dover scegliere quando e come utilizzare queste due funzioni. In particolare, si consiglia di utilizzare la funzione realloc per ridimensionare il blocco di memoria allocato precedentemente con la funzione malloc, e quindi al momento dell’inserimento di nuovi elementi all’interno dell’array.

1. Secondo esempio:

Le liste sono il più classico esempio di strutture dinamiche. In C sono molto utilizzate quelle che vengono chiamate le liste concatenate (LinkedList), utilizzate anche in altri linguaggi, come il linguaggio Java. Scrivere un programma in linguaggio C che implementi le LinkedList con i suoi metodi fondamentali (inserimento, cancellazione, ricerca e modifica).

Coaching: Una tipica difficoltà che si incontra in questo tipo di esercizi è data dallo scorrimento della lista, ed in particolare di quale istruzione abbiamo bisogno per segnalarne la fine. Si ricorda che una lista concatenata è formata da una coppia nodo-valore, in cui un nodo punta al nodo successivo. Per cui, il consiglio è verificare il valore del nodo che dovrebbe essere puntato dal precedente (NULL = la lista è finita, no altrimenti).

1. Terzo esempio:

Come detto in precedenza, gli ArrayList, a differenza delle LinkedList, sono strutture dinamiche basate su array, e quindi si hanno risultati diversi in termini di tempi di accesso e manipolazione dei dati. Gli array, infatti, essendo strutture indicizzate, sicuramente hanno un tempo di accesso ai dati minore rispetto ad un altro tipo di struttura dati non indicizzata. Ragionare su questo aspetto e provare ad implementare il seguente esercizio:

Ripetere le stesse operazioni proposte nell’esercizio 2 (inserimento, cancellazione, ricerca e modifica), provando ad implementare, questa volta, gli ArrayList.

Coaching: Gli ArrayList, come già detto, sono profondamente diversi rispetto alle liste concatenate, per via della loro struttura indicizzata. Proprio per questo motivo, quando si vanno ad implementare le operazioni di inserimento, cancellazione, modifica e ricerca, è opportuno fornire un indice intero come parametro della funzione che si andrà ad implementare.