# Test di valutazione – Modulo 1

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Carlotta |
|  |  | Cognome | Colla |
|  |  | Data | 05/02/2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile fornendo anche degli esempi.  
ATTENZIONE: Le domande a risposta multipla possono contenere più risposte corrette.

1. *Spiegare le differenze tra memoria principale, secondaria e virtuale  
   La memoria principale è una memoria volatile (se il computer si spegne i dati vengono persi), è una memoria di piccole dimensioni ma molto veloce grazie all’accesso random ai dati (ovvero, accede direttamente all’indirizzo senza scorrere quelli precedenti). Un esempio di memoria principale sono la Ram e la Cache.  
   La memoria secondaria è una memoria permanente (non si elimina allo spegnimento), è una memoria di grandi dimensioni ma l’accesso al dato è sequenziale (si scorrono tutti gli indici fino ad arrivare a quello richiesto) per questo è più lenta della memoria principale. Un esempio di memoria secondaria sono l’hard disk e i cd.  
   Sulla memoria principale vengono caricati solo i dati utili in un preciso momento (che possono essere quelli usati più frequentemente o quelli delle celle successie), questo processo è detto principio di località, grazie a questo la memoria sembra più grande di quella che è (ad esempio, può eseguire dei programmi da molti gb anche se ne può contenere circa 8). Questo aumento “apparente” di memoria viene chiamata memoria virtuale.*

1. *Cosa significa che un linguaggio è tipizzato? Spiegare la suddivisione in tipi in C# e le relative caratteristiche.  
   Un linguaggio è tipizzato quando le variabili sono suddivise in tipi di dato, i tipi indicano le caratteristiche e le funzionalità delle variabili (un esempio di tipi in c# sono int per gli interi, char per i caratteri, bool per indicare vero o falso..).  
   I tipi possono essere suddivisi in: value (per valore) e reference (per riferimento). Nei tipi value, le variabili contengono direttamente il valore, vengono salvati sullo stack ma non possono essere nulli (se non vengono inizializzati conterrano il valore di default del proprio tipo). I tipi per riferimento contengono l’indirizzo della cella di memoria dove è contenuto il dato, possono essere nulli e devono essere istanziati con new, vengono salvato nell’heap. Un esempio di tipi per riferimento sono le stringhe e le classi.*
2. *Cos’è un sistema operativo? Dare una spiegazione delle funzionalità.  
   Un sistema operativo è un insieme di software (istruzioni). Il sistema operativo fa da tramite tra i dati e quello che vede l’utente. In realtà tra il sistema operativo e i dati è presente il kernel per garantire sicurezza e multitasking.  
   Il sistema operativo deve gestire: i processi (attraverso il kernel), le interfacce utente (interfaccia bash, interfaccia vocale), coordinare le varei componenti di un computer e la gestione delle memorie (ovvero assegnare ad ogni processo un pezzo di memoria).*
3. *Descrivere il funzionamento degli operatori logici e delle relative tabelle di verità.  
   Gli operatori logici vengono utilizzati per concatenare delle espressioni logiche che possono essere vere o false. Sono due: and (&&) che ritorna vero solo se entrambe le espressioni sono vere e or (||) che ritorna vero se almeno una (o entrambe) le espressioni sono vere.  
   es. (x == y) && (y ==z) se entrambe sono vere ritorna vero.*

|  |  |  |
| --- | --- | --- |
| ***And*** | ***vero*** | ***falso*** |
| ***vero*** | *vero* | *falso* |
| ***falso*** | *falso* | *falso* |

*es. (x==y) || (y==z) se una delle due è vera )o entrambe) ritorna vero.*

|  |  |  |
| --- | --- | --- |
| ***or*** | ***vero*** | ***falso*** |
| ***vero*** | *vero* | *vero* |
| ***falso*** | *vero* | *falso* |

1. *Spiegare la differenza tra ricorsione e iterazione e le relative peculiarità.  
   La ricorsione avviene quando una funzione richiama se stessa. In questo caso, sullo stack delle chiamate, resta attiva una chiamata ad ogni ricorsione, con troppe chiamate si rischia di esaurire lo spazio dello stack, ecco perchè è più efficiente su numeri piccoli. L’iterazione è il ripetere una parte di codice più volte fino al raggiungimento del risultato desiderato. Con grandi numeri è più performante di una ricorsione.*
2. *Dare una definizione di casting. Fornire degli esempi.  
   Casting significa cambiare il tipo di una variabile in un tipo affine (es da byte a int). I casting si dividono in casting impliciti in cui non c’è perdita di memoria e si ha la sicurezza di una conversione sicura e castin espliciti in cui c’è una potenziale perdita di memoria e non si ha la certezza che la converisone vada a buon fine. Esistono molti operatori di casting, come: le parentesi tonde (es. Byte i = (Byte) j; dove j è un intero), l’operatore “is” che ritorna true o false se una variabile è di un certo tipo (es. if(x is int) ritorna true se x è converitbile in un intero), il try parse è l’operazione più sicura perchè ritorna true se il casting è andato a buon fine e il suo valore viene salvato in una nuova variabile (es Int32.tryparse(x, out int y) se x è converbile in un intero allora ritorna true e salva il valore in y, altrimenti ritona false).*

1. *Spiegare cosa è una routine. Specificare le tipologie di routine e fornire degli esempi per ciascuna.  
   Una routine è l’insieme di righe di codice che possono essere richiamate in altre parti di codice. Le routine hanno un nome e un tipo, possono avere anche dei parametri di input e dei valori di ritorno. Si suddividono in Funzioni (che hanno un tipo e un valore di ritono) e in metodi (che non hanno un tipo di ritorno, per questo sono di tipo void).  
   La firma di una routine è composta dal nome e dai parametri di input con il loro tipo di dato.*
2. *Descrivere la funzione del compilatore  
   Il compilatore traduce il codice sorgente (quello che scriviamo noi) in linguaggio macchina creando un file eseguibile. Mentre l’interprete traduce direttamente riga per riga in linguaggio macchina, senza creare il file eseguibile.*
3. *Descrivere le funzioni di un sistema di versionamento e le diverse tipologie  
   Un sistema di versionamento consente di mantenere le diverse versione di un file. Grazie a questo è possibile ritornare a una versione precedente o revisionare i cambiamenti da una verisone all’altra. Esistono tre tipi di sistemi di verisionamento: Locale (ovvero in un unico computer è presente il database delle versioni precedenti e i file che sono stati cambiati), Centralizzato (in cui sono presenti più computer in cui vengono salvati solo i file modificati e un databse esterno in cui vengono salvate le varie versioni, in questo caso se il databse si danneggi si ha la perdita delle versioni), Distribuito (in cui sono presenti più computer che salvano tutto il database delle versioni in locale e sono collegati a un database che contiene tutte le ersioni, in questo caso anche se il database si danneggia non si ha la paerdita delle versioni).*

*Esercizio Pratico*

Creare una Console Application che gestisca i Task dell’utente.

Per Task viene inteso un oggetto che ha una descrizione, una data di scadenza e un livello di importanza (Basso, Medio, Alto).

L’utente può:

* Vedere i Task inseriti
* Aggiungere un nuovo Task
* Eliminare un Task
* Filtrare i Task per importanza

Requisiti Tecnici:

-Salvare i Task in un file

-Utilizzare adeguatamente il concetto di classe

-Dividere le funzionalità in relative funzioni e procedure

-Commentare

-Mettere una nomenclatura conforme

-Le date di scadenza devono essere posteriori o uguali rispetto alla data di inserimento

-Controllare l’input utente

Opzionale: Utilizzare Enum

Mettere il codice dell’esercizio in un Repository di GitHub.