Programmazione I
Prova scritta - 21 gennaio 2015 - 1h20min

NOTA: Nei programmi si trascuri ogni problema legato al tipo ed al valore di ritorno della funzione **main**, inoltre si sottintenda la presenza delle direttive
`#include <iostream>` / `#include <fstream>` / `using namespace std ;`
e non si prenda come un buon esempio la formattazione utilizzata (spesso compressa per motivi di spazio). Si interpreti "terminazione forzata", come l'abbreviazione di "terminazione forzata del programma da parte del sistema operativo". Infine, laddove si trovi l'affermazione che un programma o frammento di codice produce un certo risultato, è da intendersi che, in accordo alle regole del linguaggio, tale programma o frammento di codice produce quel risultato per **qualsiasi esecuzione** su **qualsiasi macchina**.

PARTE 1 – RISPOSTA SINGOLA - Ogni domanda ha una sola risposta VERA.

- **Una risposta esatta fa acquisire il punteggio positivo riportato a fianco della domanda**
- **Una risposta errata fa perdere il punteggio negativo riportato a fianco della domanda**
- **Una risposta lasciata in bianco viene valutata 0**

1. **(3, -0.5)** Dato il seguente programma:

```
int *a, num_elem ;
void distruggi() { delete [] a; num_elem = 0 ; }
void inserisci(int n) {a[num_elem] = n ; num_elem++ ;}
void stampa() { for (int i = 0 ; i < num_elem ; i++) cout<<a[i]<<" " ; }

main(){
    a = new int[10] ;
    inserisci(2) ; stampa() ;
    distruggi() ;
    a = new int[5] ;
    inserisci(3) ; stampa() ;
}
```

   a) il programma stampa `2 3`
   b) nessuna delle altre risposte è vera
   c) il programma potrebbe essere terminato forzatamente all'atto della seconda allocazione dinamica della memoria, a causa di un errore di gestione della memoria
   d) il programma contiene un errore di gestione della memoria diverso dal *memory leak*

2. **(3, -0.5)** Dato il seguente programma:

```
int fun(const int a[]) { return a[0] + 1 ; }
main() { int b[2] = {2, 3} ; cout<<fun(b) ; }
```

   a) Quando la funzione **fun** è invocata, il contenuto dell'*array* **b** è copiato nell'*array* **a**
   b) Quando la funzione **fun** è invocata, l'indirizzo (del primo elemento) dell'*array* **b** è copiato nel parametro formale **a**
   c) Se la funzione **fun** contenesse istruzioni che modificano il valore di un elemento dell'*array* **a** (parametro formale) sarebbe segnalato un errore a tempo di esecuzione, in particolare l'errore sarebbe segnalato nel momento in cui si tenta di eseguire tale istruzione
   d) Nessuna delle altre risposte è vera

3. **(3, -0.5)** Dato il seguente programma, e ricordando che il tipo **unsigned int** è gerarchicamente superiore al tipo **int**,

```
main() {
 int i ; cin>>i ; unsigned int u = 10 ;
 cout<<(u+i) ;
}
```

a) per qualsiasi numero intero, positivo o negativo, immesso dall'utente, il programma stampa un numero non negativo
b) se l'utente immette `-10` da *stdin*, il programma viene terminato forzatamente
c) se l'utente immette `-20` da *stdin*, il programma stampa `-10`
d) nessuna delle altre risposte è vera

4. **(2, -0.5)** Il seguente frammento di codice:

```
for (int i=4; i > 0 ; i--) { if (i==3) i = 2; else cout<<i*2<<" ";}
```

a) Stampa `8 2`
b) Contiene un ciclo infinito
c) Nessuna delle altre risposte è vera
d) Stampa `8 4 2`

PARTE 2 – (POSSIBILI) RISPOSTE MULTIPLE - Ogni domanda può avere da una a quattro risposte CORRETTE.

- **Ogni risposta esatta viene calcolata: +1**
- **Ogni risposta errata viene calcolata: -0.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. Dato un programma scritto in linguaggio C/C++
   a) Per deallocare un record di attivazione è necessario reinizializzare il contenuto delle celle di memoria precedentemente occupate dal record stesso
   b) Il tempo necessario per inizializzare il contenuto di un record di attivazione aumenta all'aumentare del numero e delle dimensioni dei parametri formali
   c) Il record di attivazione di ogni funzione contiene (oltre ad altre informazioni) le variabili globali a cui accede la funzione
   d) Il tempo necessario per inizializzare il contenuto di un record di attivazione aumenta all'aumentare del numero e delle dimensioni delle variabili locali inizializzate

6. Dato il seguente frammento di codice contenuto all'interno di una funzione **main**:

```
struct st {
   char a[20];
   int b;
} ;
st s, *ptr = 0 ;
```

quale/i delle seguenti istruzioni, se eseguite immediatamente dopo tale frammento, è/sono corretta/e?

a) `strcpy(ptr->a, "Testo");`
b) `s.b=10;`
c) `ptr = &s;`
d) `ptr = s;`

7. Quali delle seguenti affermazioni sono vere?
   a) Un file binario può contenere sequenze di byte uguali a quelle che possono essere contenute in un file di testo
   b) Un file di testo non contiene numeri, ma rappresentazioni grafiche di caratteri (alcuni dei quali possono eventualmente essere cifre numeriche)
   c) Nessuna delle altre risposte è vera
   d) Se si memorizza un array di caratteri in un file mediante scrittura non formattata, allora il file risultante è un file di testo contenente la sequenza dei codici dei caratteri contenuti nell'array

8. Data una sequenza di N elementi tutti dello stesso tipo
   a) Se la sequenza è memorizzata in *array* e si vuole leggere il valore di un elemento di cui si conosce solo la posizione nella sequenza (si sa che è l'elemento i-esimo, con i<N), allora la lettura può essere effettuata a costo O(1)
   b) Se la sequenza è memorizzata in una lista semplice e si vuole leggere il valore di un elemento di cui si conosce solo la posizione nella sequenza (si sa che è l'elemento i-esimo, con i<N), allora la lettura può essere effettuata a costo O(1)
   c) Se la sequenza è memorizzata in un *array* utilizzando un terminatore per delimitare la fine della sequenza stessa, non è necessario mantenere ulteriori informazioni per poter accedere sequenzialmente a tutti gli elementi della sequenza
   d) A seconda del tipo degli elementi (uguale per tutti gli elementi) della sequenza e del valore di N, memorizzare la sequenza in una lista può essere più efficiente di memorizzare, in termini di occupazione della memoria, la sequenza in un *array* con terminatore (ossia un array in cui si utilizza un terminatore per delimitare la fine della sequenza)

9. Dato il seguente programma e facendo attenzione alle costanti letterali utilizzate:

```
main() {
   char c ; cin>>c ;
   switch(c) {
   case 0:
        cout<<"Primo" ;
   case '3':
        cout<<"Secondo" ;
        break ;
   default:
        cout<<"Errore" ;
   }
}
```

   a) Se l'utente immette le cifre del codice numerico usato per rappresentare il carattere **3** su *stdin*, il programma stampa **Secondo**
   b) Il programma causa un errore a tempo di compilazione perché non si può utilizzare il tipo *char* in uno *switch*
   c) Se l'utente immette il carattere **0** su *stdin*, il programma stampa **Errore**
   d) Se l'utente immette il carattere **3** su *stdin*, il programma stampa **Secondo**

PARTE 3 – DOMANDE APERTE

- **Una risposta esatta fa acquisire il punteggio positivo riportato a fianco della domanda**
- **Una risposta errata può eventualmente causare una penalità che dipende dalla gravità dell'errore**
- **Una risposta lasciata in bianco viene calcolata: 0**

10. **(6 pt)** Descrivere sintassi, semantica e vantaggi del tipo *booleano*, in non più di otto righe, più eventuali frammenti di codice o ulteriori righe scritte in una qualsiasi notazione (a supporto della descrizione della sintassi).

11. **(5 pt)** Scrivere una funzione che prende in ingresso:
    1. una sequenza di operazioni, di lunghezza massima fissata a tempo di scrittura del programma, in cui ciascuna operazione è caratterizzata da un *codice operazione* (numero intero) ed un *intervallo di tempo* (numero intero positivo), uguale al tempo che deve trascorrere prima che l'operazione sia eseguita, a partire dall'istante in cui l'ultima operazione è stata eseguita, o a partire dall'istante corrente nel caso della prima operazione;
    2. il codice della prossima operazione da eseguire e l'intervallo di tempo da aspettare prima di eseguirla;

ed inserisce tale operazione in fondo alla sequenza, gestendo le possibili situazioni di errore (non è richiesto di realizzare anche una funzione che estrae gli elementi dalla sequenza). Ad esempio, se si passano in ingresso alla funzione la sequenza [(45, 3), (-3, 5)], il codice 7 e l'intervallo 9, la funzione trasforma la sequenza in [(45, 3), (-3, 5), (7, 9)]. Definire inoltre esplicitamente anche la struttura dati con cui la sequenza è implementata.

Programmazione I
Prova scritta - 21 gennaio 2015

Nome: _____ Cognome: _____

Matricola: _____

Indicare le risposte corrette apponendo una croce nella casella corrispondente. Per superare la prova bisogna aver raggiunto almeno 9 punti nelle domande a risposta singola/multipla, ed almeno 15 complessivamente. Questa è l'unica pagina che dovete consegnare. Per comodità avete anche una copia di questa pagina per calcolare il voto da sole/soli durante la correzione.

	Risposte				Punti/ Penalità	
	A	B	C	D		
1					3	-0,5
2					3	-0,5
3					3	-0,5
4					2	-0,5
5						
6						
7						
8						
9						

Risposta alla domanda 10 (6 pt):

Risposta alla domanda 11 (5 pt):

Programmazione I
Prova scritta - 21 gennaio 2015

Nome: _____ Cognome: _____

Matricola: _____

Indicare le risposte corrette apponendo una croce nella casella corrispondente. Per superare la prova bisogna aver raggiunto almeno 9 punti nelle domande a risposta singola/multipla, ed almeno 15 complessivamente. Usate questa copia per calcolare il voto da sole/soli durante la correzione.

	Risposte				Punti/ Penalità	
	A	**B**	**C**	**D**		
1					**3**	**-0,5**
2					**3**	**-0,5**
3					**3**	**-0,5**
4					**2**	**-0,5**
5						
6						
7						
8						
9						

Risposta alla domanda 10 (6 pt):

Risposta alla domanda 11 (5 pt):