Programmazione I
Prova scritta - 9 gennaio 2017 - 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)** Supponendo che la matrice `int a[2][5]` sia memorizzata a partire dall'indirizzo 100 e che gli oggetti di tipo `int` occupino 4 byte, l'indirizzo dell'elemento `a[1][2]` è
   a) 100 + 0*4*5 + 2*4 = 108
   b) 100 + 1*4*2 + 1*4 = 112
   c) 100 + 1*4*5+ 2*4 = 128
   d) nessuna delle altre risposte è corretta

2. **(3, -.5)** Assumendo che i valori di tipo **double** siano memorizzati in accordo allo standard IEEE 754 (in base 2):
   a) Il tipo **double** può rappresentare tutti i numeri reali compresi nel suo intervallo di rappresentabilità
   b) La precisione del tipo **double** dipende dal numero di cifre riservate alla rappresentazione dell'esponente
   c) Se **a**, **b** e **c** sono tre variabili di tipo double, **a** e **b** contengono due valori positivi, e si esegue l'assegnamento **c = a + b**, senza che la somma **a + b** generi *overflow*, allora, dopo tale assegnamento, la seguente espressione logica è vera: **a == c – b**.
   d) Nessuna delle altre risposte è vera

3. **(2, -0.5)** La seguente istruzione
   `int b = static_cast<double>(5)/2+3./2;`
   fornisce:
   a) un errore a tempo di compilazione perché il risultato dell'espressione non è rappresentabile mediante il tipo **int**
   b) il valore 4 nella variabile **b**
   c) una terminazione forzata a tempo di esecuzione perché il risultato dell'espressione non è rappresentabile mediante il tipo **int**
   d) il valore 3 nella variabile **b**

4. **(3, -.5)** Dato il seguente programma e supponendo che l'operatore >> non modifichi il valore della variabile se l'oggetto **cin** è oppure entra in stato di errore:

```
main()
{
 int i = 10 ;
 do {cin>>i ;
   cout<<i<<", " ; }
 while(cin) ;
}
```

se l'utente inserisce da *stdin* i caratteri `1` e `0`, separati da uno spazio, quindi preme *Invio*, ed infine preme CTRL+D (su sistema UNIX)

a) il programma stampa su *stdout* `1, 0,`
b) il programma stampa su *stdout* `1, 0, 10,`
c) il programma stampa su *stdout* `1, 0, 0,`
d) nessuna delle altre risposte è corretta

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: -.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. Due algoritmi equivalenti:
   a) Forniscono lo stesso risultato per qualsiasi insieme di dati in ingresso
   b) Hanno lo stesso codice
   c) Prevedono gli stessi passi
   d) A parità di dati in ingresso possono avere tempi di esecuzione diversi

6. Dato il seguente programma:

```
 1:float b = 3.5;
 2:float fun(float a)
 3:{
 4:     int i ;
 5:     for (i = 0 ; i < 2 ; i++)
 6:        a *= 2 ;
 7:     return a + b + i ;
 8:}
 9:
10:main()
11:{
12:    float b = 1 ;
13:    float c = fun(b) ;
14:    cout<<static_cast<int>(c)<<endl ;
15:}
```

a) Quando eseguita, la funzione `fun` non modifica il valore della variabile `b` definita alla riga 12
b) Il parametro formale `a` definito alla riga 2 non è visibile alla riga 14
c) La variabile `i` definita alla riga 4 non è visibile alla riga 7
d) Il programma stampa `9`

7. Dato il seguente programma:

```
const int N = 100 ; struct ss { int b; char a[N] ; } ;
void fun(ss & d) { cout<<d.a[N/2]<<" "; d.b = 4 ; }
main()
{
   ss c ; c.b = 7 ; c.a[N/2] = '2' ; cout<<c.b<<" " ;
   fun(c) ; cout<<c.b ;
}
```

   a) Il passaggio di **c** per riferimento permette a **fun** di modificare **c**
   b) Il programma stampa **7 2 4**
   c) Il passaggio di **c** per riferimento ha come costo la copia nel parametro formale **d** del contenuto di **c** stesso
   d) L'esecuzione di **fun(c)** ha effetti collaterali sulla porzione della funzione **main** che segue l'invocazione della funzione **fun** stessa

8. Dato un programma scritto in linguaggio C/C++
   a) Il tempo necessario per inizializzare il contenuto di un record di attivazione è indipendente dal numero e le dimensioni delle variabili locali inizializzate
   b) Il record di attivazione relativo ad una data funzione non contiene le variabili globali a cui accede la funzione
   c) Il record di attivazione contiene gli indirizzi degli eventuali parametri attuali passati per riferimento
   d) Per deallocare correttamente un record di attivazione è necessario reinizializzare il contenuto dei campi che tale record contiene

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**

9. **(5 pt)** Descrivere la motivazione per l'uso di array dinamici e gli svantaggi dell'uso di array dinamici rispetto ad array automatici, in non più di 8 righe (non verrà valutata la quantità ma la qualità di quello che si scrive, ed il tentativo di scrivere in modo estremamente fitto per aumentare la quantità, così come il superamento del numero massimo di righe, comporteranno una penalità).

10. **(7 pt)** Scrivere una funzione che prende in ingresso due stringhe, **a** e **b**, e, se non vi è alcuna occorrenza della stringa stringa **b** all'interno della stringa **a**, allora aggiunge la stringa **b** in fonda alla stringa **a**. Ad esempio, se alla funzione vengono passate le stringhe *"Piu' per meno "* (in questo esempio c'è uno spazio in fondo alla stringa) e *"meno"*, allora la funzione non modifica la stringa **a**. Se invece la stringa **a** è la stessa del precedente esempio, ma la stringa **b** è *"Meno"*, allora trasforma la prima stringa in *"Piu' per meno Meno"*.

Programmazione I
Prova scritta - 6 febbraio 2017

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					**2**	**-0,5**
4					**3**	**-0,5**
5						
6						
7						
8						

Risposta alla domanda 9 (5 pt):

Risposta alla domanda 10 (7 pt):

Programmazione I
Prova scritta - 6 febbraio 2017

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

Risposta alla domanda 9 (5 pt):

Risposta alla domanda 10 (7 pt):