Programmazione I
Prova scritta - 6 Febbraio 2019 - 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. **(2, -.5)** Il seguente ciclo:

```
int i = 2;
while(i <= 10) {
      if (i % 10 == 0) i/=2 ;
      else {
            cout<<i<<endl;
            i*=2 ;
      }
}
```

a) produce un errore a tempo di esecuzione;
b) va avanti all'infinito;
c) termina dopo un numero finito di passi;
d) non stampa nulla.

2. **(2, -.5)** Dato un file sorgente C/C++ contenente la seguente direttiva

```
#define COSTANTE1 15
```

a) tale direttiva è perfettamente equivalente all'istruzione

```
const int COSTANTE1 = 15 ;
```

b) il preprocessore sostituisce testualmente ogni occorrenza della parola **COSTANTE1** con la sequenza di due caratteri `15`;
c) la parola `COSTANTE1` è ancora presente nel file prodotto dal preprocessore e preso in ingresso per la traduzione;
d) la presenza di tale direttiva e della seguente istruzione
`int COSTANTE1 ;`
all'interno di una qualche funzione definita in un punto successivo del file (successivo rispetto a quello in cui è presente la direttiva) non comporta errori di compilazione.

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

```
void fun(int a[])
{ int n; cin>>n ; for (int i = 0 ; i < n ; i++) cin>>a[i] ;}

main() { int c[3] ; fun(c) ; }
```

a) Il tempo di vita dell'*array* **c** non include l'intervallo di tempo durante il quale la funzione **fun** è eseguita
b) La funzione **fun** modifica solo una copia locale dell'*array* **c** che le viene passato
c) La variabile **n** ha classe di memorizzazione automatica
d) Nessuna delle altre risposte è corretta

4. **(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

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

- **Ogni risposta esatta viene calcolata: +1**
- **Ogni risposta errata viene calcolata: -.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. 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 = 2.5 ;
13:   float c = fun(b) ;
14:   cout<<static_cast<int>(c + b)<<endl ;
15:}
```

a) la funzione **`fun`** 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 **`b`** definita alla riga 1 è visibile solo nella funzione **`fun`**
d) il programma stampa **25**

6. Dato un array così dichiarato
```
int a[5] ;
```
all'interno di una funzione

a) l'istruzione **`a[5] = 5 ;`** non è corretta;
b) l'istruzione **`a[-1] = 0 ;`** è corretta ed assegna il valore 0 all'elemento che precede il primo elemento dell'array;
c) l'istruzione **`a[0]=-1 ;`** è corretta ed assegna il valore -1 al primo elemento dell'array.
d) l'espressione **`a`** può essere utilizzata per passare l'array ad una funzione;

7. L'istruzione
```
ifstream f("DATI") ;
```

a) apre un file di nome **DATI** in lettura
b) se l'operazione di apertura del file va a buon fine è legale l'esecuzione delle istruzioni
```
{ int data; cin >> data;  f << data; }
```
c) può avere successo solo se il file da aprire è di tipo binario
d) se il file **DATI** non esiste, non lo crea

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**

8. **(7 pt)** Scrivere una funzione che prenda in ingresso due numeri interi positivi **N** ed **M**, e crei e ritorni un array di **M** numeri interi positivi. Se la rappresentazione in base **10** del numero **N** ha al più **M** cifre, la funzione inserisce in ciascun elemento dell'array una delle cifre di tale rappresentazione. In particolare, la cifra di peso minimo del numero **N** (le unità) viene inserita nell'ultimo elemento dell'array, quella di peso successivo (le decine) nel penultimo elemento, e così via. Gli elementi dell'array a cui non corrisponde nessuna cifra sono riempiti col valore 0. Esempio: se **N** = 436, **M** = 5, la funzione ritorna l'array: {0, 0, 4, 3, 6}. Se il numero **N** ha più di **M** cifre la funzione termina senza compiere alcuna operazione.

9. **(6 pt)** Descrivere sintassi, semantica e vantaggi del tipo enumerato, in non più di otto righe più eventuali frammenti di codice o righe scritte in una qualche notazione (a supporto della descrizione della sintassi).

Programmazione I
Prova scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
Prova scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
Prova Scritta - 6 Febbraio 2019 - 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. (2, -.5) Dato un file sorgente C/C++ contenente la seguente direttiva

```
#define COSTANTE1 15
```

a) il preprocessore sostituisce testualmente ogni occorrenza della parola **COSTANTE1** con la sequenza di due caratteri `15`;
b) la parola `COSTANTE1` è ancora presente nel file prodotto dal preprocessore e preso in ingresso per la traduzione;
c) la presenza di tale direttiva e della seguente istruzione
`int COSTANTE1 ;`
all'interno di una qualche funzione definita in un punto successivo del file (successivo rispetto a quello in cui è presente la direttiva) non comporta errori di compilazione.
d) tale direttiva è perfettamente equivalente all'istruzione

```
const int COSTANTE1 = 15 ;
```

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

```
void fun(int a[])
{ int n; cin>>n ; for (int i = 0 ; i < n ; i++) cin>>a[i] ;}

main() { int c[3] ; fun(c) ; }
```

a) La funzione **fun** modifica solo una copia locale dell'*array* **c** che le viene passato
b) La variabile **n** ha classe di memorizzazione automatica
c) Nessuna delle altre risposte è corretta
d) Il tempo di vita dell'*array* **c** non include l'intervallo di tempo durante il quale la funzione **fun** è eseguita

3. **(3, -.5)** Assumendo che i valori di tipo **double** siano memorizzati in accordo allo standard IEEE 754 (in base 2):
   a) La precisione del tipo **double** dipende dal numero di cifre riservate alla rappresentazione dell'esponente
   b) 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**.
   c) Nessuna delle altre risposte è vera
   d) Il tipo **double** può rappresentare tutti i numeri reali compresi nel suo intervallo di rappresentabilità

4. (2, -.5) Il seguente ciclo:

```
int i = 2;
while(i <= 10) {
      if (i % 10 == 0) i/=2 ;
      else {
            cout<<i<<endl;
            i*=2 ;
      }}
```

   a) va avanti all'infinito;
   b) termina dopo un numero finito di passi;
   c) non stampa nulla.
   d) produce un errore a tempo di esecuzione;

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

- **Ogni risposta esatta viene calcolata: +1**
- **Ogni risposta errata viene calcolata: -.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. Dato un array così dichiarato
   `int a[5] ;`
   all'interno di una funzione

   a) l'istruzione `a[-1] = 0 ;` è corretta ed assegna il valore 0 all'elemento che precede il primo elemento dell'array;
   b) l'istruzione `a[0]=-1 ;` è corretta ed assegna il valore -1 al primo elemento dell'array.
   c) l'espressione `a` può essere utilizzata per passare l'array ad una funzione;
   d) l'istruzione `a[5] = 5 ;` non è corretta;

7. L'istruzione
   `ifstream f("DATI") ;`

   a) se l'operazione di apertura del file va a buon fine è legale l'esecuzione delle istruzioni
      `{ int data; cin >> data;  f << data; }`
   b) può avere successo solo se il file da aprire è di tipo binario
   c) se il file **DATI** non esiste, non lo crea
   d) apre un file di nome **DATI** in lettura

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 = 2.5 ;
13:   float c = fun(b) ;
14:   cout<<static_cast<int>(c + b)<<endl ;
  1  }
```

a) il parametro formale **a** definito alla riga 2 non è visibile alla riga 14
b) la variabile **b** definita alla riga 1 è visibile solo nella funzione **fun**
c) il programma stampa **25**
d) la funzione **fun** modifica il valore della variabile **b** definita alla riga 12

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**

8. **(7 pt)** Scrivere una funzione che prenda in ingresso due numeri interi positivi **N** ed **M**, e crei e ritorni un array di **M** numeri interi positivi. Se la rappresentazione in base **10** del numero **N** ha al più **M** cifre, la funzione inserisce in ciascun elemento dell'array una delle cifre di tale rappresentazione. In particolare, la cifra di peso minimo del numero **N** (le unità) viene inserita nell'ultimo elemento dell'array, quella di peso successivo (le decine) nel penultimo elemento, e così via. Gli elementi dell'array a cui non corrisponde nessuna cifra sono riempiti col valore 0. Esempio: se **N** = 436, **M** = 5, la funzione ritorna l'array: {0, 0, 4, 3, 6}. Se il numero **N** ha più di **M** cifre la funzione termina senza compiere alcuna operazione.

9. **(6 pt)** Descrivere sintassi, semantica e vantaggi del tipo enumerato, in non più di otto righe più eventuali frammenti di codice o righe scritte in una qualche notazione (a supporto della descrizione della sintassi).

Programmazione I
Prova Scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
Prova Scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
prova Scritta - 6 Febbraio 2019 - 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:

```
void fun(int a[])
{ int n; cin>>n ; for (int i = 0 ; i < n ; i++) cin>>a[i] ;}

main() { int c[3] ; fun(c) ; }
```

a) La variabile **n** ha classe di memorizzazione automatica
b) Nessuna delle altre risposte è corretta
c) Il tempo di vita dell'*array* **c** non include l'intervallo di tempo durante il quale la funzione **fun** è eseguita
d) La funzione **fun** modifica solo una copia locale dell'*array* **c** che le viene passato

2. **(3, -.5)** Assumendo che i valori di tipo **double** siano memorizzati in accordo allo standard IEEE 754 (in base 2):
   a) 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**.
   b) Nessuna delle altre risposte è vera
   c) Il tipo **double** può rappresentare tutti i numeri reali compresi nel suo intervallo di rappresentabilità
   d) La precisione del tipo **double** dipende dal numero di cifre riservate alla rappresentazione dell'esponente

3. **(2, -.5)** Il seguente ciclo:

```
int i = 2;
while(i <= 10) {
      if (i % 10 == 0) i/=2 ;
      else {
            cout<<i<<endl;
            i*=2 ;
      }}
```

a) termina dopo un numero finito di passi;
b) non stampa nulla.
c) produce un errore a tempo di esecuzione;
d) va avanti all'infinito;

4. **(2, -.5)** Dato un file sorgente C/C++ contenente la seguente direttiva

```
#define COSTANTE1 15
```

a) la parola **COSTANTE1** è ancora presente nel file prodotto dal preprocessore e preso in ingresso per la traduzione;
b) la presenza di tale direttiva e della seguente istruzione
`int COSTANTE1 ;`
all'interno di una qualche funzione definita in un punto successivo del file (successivo rispetto a quello in cui è presente la direttiva) non comporta errori di compilazione.
c) tale direttiva è perfettamente equivalente all'istruzione

```
const int COSTANTE1 = 15 ;
```

d) il preprocessore sostituisce testualmente ogni occorrenza della parola **COSTANTE1** con la sequenza di due caratteri **15**;

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

- **Ogni risposta esatta viene calcolata: +1**
- **Ogni risposta errata viene calcolata: -.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. L'istruzione

```
ifstream f("DATI") ;
```

a) può avere successo solo se il file da aprire è di tipo binario
b) se il file **DATI** non esiste, non lo crea
c) apre un file di nome **DATI** in lettura
d) se l'operazione di apertura del file va a buon fine è legale l'esecuzione delle istruzioni
`{ int data; cin >> data;  f << data; }`

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 = 2.5 ;
13:   float c = fun(b) ;
14:   cout<<static_cast<int>(c + b)<<endl ;
15:}
```

a) la variabile **b** definita alla riga 1 è visibile solo nella funzione **fun**
b) il programma stampa **25**
c) la funzione **fun** modifica il valore della variabile **b** definita alla riga 12
d) il parametro formale **a** definito alla riga 2 non è visibile alla riga 14

7. Dato un array così dichiarato

```
int a[5] ;
```

all'interno di una funzione

a) l'istruzione `a[0]=-1 ;` è corretta ed assegna il valore -1 al primo elemento dell'array.
b) l'espressione **a** può essere utilizzata per passare l'array ad una funzione;
c) l'istruzione `a[5] = 5 ;` non è corretta;
d) l'istruzione `a[-1] = 0 ;` è corretta ed assegna il valore 0 all'elemento che precede il primo elemento dell'array;

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**

8. **(7 pt)** Scrivere una funzione che prenda in ingresso due numeri interi positivi **N** ed **M**, e crei e ritorni un array di **M** numeri interi positivi. Se la rappresentazione in base **10** del numero **N** ha al più **M** cifre, la funzione inserisce in ciascun elemento dell'array una delle cifre di tale rappresentazione. In particolare, la cifra di peso minimo del numero **N** (le unità) viene inserita nell'ultimo elemento dell'array, quella di peso successivo (le decine) nel penultimo elemento, e così via. Gli elementi dell'array a cui non corrisponde nessuna cifra sono riempiti col valore 0. Esempio: se **N** = 436, **M** = 5, la funzione ritorna l'array: {0, 0, 4, 3, 6}. Se il numero **N** ha più di **M** cifre la funzione termina senza compiere alcuna operazione.

9. **(6 pt)** Descrivere sintassi, semantica e vantaggi del tipo enumerato, in non più di otto righe più eventuali frammenti di codice o righe scritte in una qualche notazione (a supporto della descrizione della sintassi).

Programmazione I
prova Scritta – 6 Febbraio 2019

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					**2**	**-0,5**
5						
6						
7						

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
prova Scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
prova scritta - 6 Febbraio 2019 - 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, -.5)** Assumendo che i valori di tipo **double** siano memorizzati in accordo allo standard IEEE 754 (in base 2):
   a) Nessuna delle altre risposte è vera
   b) Il tipo **double** può rappresentare tutti i numeri reali compresi nel suo intervallo di rappresentabilità
   c) La precisione del tipo **double** dipende dal numero di cifre riservate alla rappresentazione dell'esponente
   d) 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**.

2. **(2, -.5)** Il seguente ciclo:

```
int i = 2;
while(i <= 10) {
     if (i % 10 == 0) i/=2 ;
     else {
          cout<<i<<endl;
          i*=2 ;
     }}
```

   a) non stampa nulla.
   b) produce un errore a tempo di esecuzione;
   c) va avanti all'infinito;
   d) termina dopo un numero finito di passi;

3. (2, -.5) Dato un file sorgente C/C++ contenente la seguente direttiva

```
#define COSTANTE1 15
```

a) la presenza di tale direttiva e della seguente istruzione
`int COSTANTE1 ;`
all'interno di una qualche funzione definita in un punto successivo del file (successivo rispetto a quello in cui è presente la direttiva) non comporta errori di compilazione.

b) tale direttiva è perfettamente equivalente all'istruzione

```
const int COSTANTE1 = 15 ;
```

c) il preprocessore sostituisce testualmente ogni occorrenza della parola **COSTANTE1** con la sequenza di due caratteri `15`;

d) la parola `COSTANTE1` è ancora presente nel file prodotto dal preprocessore e preso in ingresso per la traduzione;

4. (3, -0.5) Dato il seguente programma:

```
void fun(int a[])
{ int n; cin>>n ; for (int i = 0 ; i < n ; i++) cin>>a[i] ;}

main() { int c[3] ; fun(c) ; }
```

a) Nessuna delle altre risposte è corretta
b) Il tempo di vita dell'*array* **c** non include l'intervallo di tempo durante il quale la funzione **fun** è eseguita
c) La funzione **fun** modifica solo una copia locale dell'*array* **c** che le viene passato
d) La variabile **n** ha classe di memorizzazione automatica

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

- **Ogni risposta esatta viene calcolata: +1**
- **Ogni risposta errata viene calcolata: -.5**
- **Una risposta lasciata in bianco viene calcolata: 0**

5. 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() {
12:   float b = 2.5 ;
13:   float c = fun(b) ;
14:   cout<<static_cast<int>(c + b)<<endl ; }
```

a) il programma stampa **25**
b) la funzione **`fun`** modifica il valore della variabile **`b`** definita alla riga 12
c) il parametro formale **`a`** definito alla riga 2 non è visibile alla riga 14
d) la variabile **`b`** definita alla riga 1 è visibile solo nella funzione **`fun`**

6. Dato un array così dichiarato

```
int a[5] ;
```

all'interno di una funzione

a) l'espressione **`a`** può essere utilizzata per passare l'array ad una funzione;
b) l'istruzione **`a[5] = 5 ;`** non è corretta;
c) l'istruzione **`a[-1] = 0 ;`** è corretta ed assegna il valore 0 all'elemento che precede il primo elemento dell'array;
d) l'istruzione **`a[0]=-1 ;`** è corretta ed assegna il valore -1 al primo elemento dell'array.

7. L'istruzione

```
ifstream f("DATI") ;
```

a) se il file **DATI** non esiste, non lo crea
b) apre un file di nome **DATI** in lettura
c) se l'operazione di apertura del file va a buon fine è legale l'esecuzione delle istruzioni
```
{ int data; cin >> data;  f << data; }
```
d) può avere successo solo se il file da aprire è di tipo binario

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**

8. **(7 pt)** Scrivere una funzione che prenda in ingresso due numeri interi positivi **N** ed **M**, e crei e ritorni un array di **M** numeri interi positivi. Se la rappresentazione in base **10** del numero **N** ha al più **M** cifre, la funzione inserisce in ciascun elemento dell'array una delle cifre di tale rappresentazione. In particolare, la cifra di peso minimo del numero **N** (le unità) viene inserita nell'ultimo elemento dell'array, quella di peso successivo (le decine) nel penultimo elemento, e così via. Gli elementi dell'array a cui non corrisponde nessuna cifra sono riempiti col valore 0. Esempio: se **N** = 436, **M** = 5, la funzione ritorna l'array: {0, 0, 4, 3, 6}. Se il numero **N** ha più di **M** cifre la funzione termina senza compiere alcuna operazione.

9. **(6 pt)** Descrivere sintassi, semantica e vantaggi del tipo enumerato, in non più di otto righe più eventuali frammenti di codice o righe scritte in una qualche notazione (a supporto della descrizione della sintassi).

Programmazione I
prova scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):

Programmazione I
prova scritta – 6 Febbraio 2019

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

Risposta alla domanda 8 (7 pt):

Risposta alla domanda 9 (6 pt):