FONDAMENTI DI PROGRAMMAZIONE B

Tempo a disposizione: 30 minuti

Nome Aurora ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. [C++] Si consideri il seguente programma

```
void f() {
  throw 1;
}

int main() {
  try {
    f();
  } catch(int x) {
    cout << "1" << endl;
  } catch(string y) {
    cout << "2" << endl;
  } catch(Razionale z) {
    cout << "3" << endl;
  }
  return 0;
}
```

Si indichi cosa viene stampato a video dalla funzione `main`.

[ ] *a* `1 2` [X] *b* `1` [ ] *c* `1 2 3` [ ] *d* nessuna delle precedenti

2. [C++] Si considerino le classi `Forma`, `Rettangolo` e `Quadrato`. La classe `Rettangolo` è derivata da `Forma`, mentre la classe `Quadrato` è derivata da `Rettangolo`. La seguente funzione


```
double shift(Rettangolo r) {...}
```

può accettare come argomenti oggetti

[ ] *a* esclusivamente di tipo `Rettangolo` e delle sue superclassi
[ ] *b* di tipo `Forma`, `Rettangolo` e `Object` ma non `Quadrato`
[ ] *c* di tipo `Forma` e `Quadrato`
[X] *d* di tipo `Rettangolo` e `Quadrato`
[ ] *e* nessuna delle precedenti

3. [C++] È possibile allocare gli oggetti esclusivamentre tramite allocazione dinamica [ ] T [X] F

4. [C++] Si consideri il template di classe `Stack⟨T⟩`. Allora `Stack⟨int⟩` è una classe derivata da `Stack⟨float⟩` [ ] T [X] F

5. [C++] Una classe `C` ha a disposizione il distruttore esclusivamente se è stato implementato un costruttore. [ ] T [X] F

6. [**Java**] Si considerino le classi `Forma`, `Triangolo` e `Quadrato`. Le classi `Quadrato` e `Triangolo` sono derivate da `Forma`. La classe `Forma` definisce un metodo `perimetro` che le classi `Quadrato` e `Triangolo` ridefiniscono. Si consideri il seguente frammento di codice.

```
Forma f = new Triangolo();
((Triangolo) f).perimetro();
```

- [ ] *a* viene sollevata una `ClassCastException`
- [ ] *b* viene rilevato un errore a tempo di compilazione
- [x] *c* viene invocato il metodo `perimetro` definito nella classe `Triangolo`
- [ ] *d* viene invocato il metodo `perimetro` definito nella classe `Forma`
- [ ] *e* nessuna delle precedenti

7. [**Java**] Si consideri la seguente dichiarazione di attributo all'interno di una classe `C`:

```
private static final int x = 1;
```

Si indichi la risposta corretta.

- [ ] *a* è un'attributo di classe con visibilità privata e modificabile
- [ ] *b* è un'attributo d'istanza con visibilità privata e non modificabile
- [x] *c* è un'attributo di classe con visibilità privata e non modificabile
- [ ] *d* è un'attributo di classe con visibilità di package e modificabile
- [ ] *e* nessuna delle precedenti

8. [**Java**] Si considerino le classi `Forma`, `Rettangolo` e `Quadrato`. La classe `Rettangolo` è derivata da `Forma`, mentre la classe `Quadrato` è derivata da `Rettangolo`. Le classi di appartenenza della classe `Rettangolo` sono

- [ ] *a* `Forma` e `Object`
- [x] *b* `Rettangolo`, `Forma` e `Object`
- [ ] *c* `Rettangolo`, `Quadrato`, `Forma` e `Object`
- [ ] *d* `Rettangolo`, `Forma`, `Quadrato`
- [ ] *e* nessuna delle precedenti

9. [**Java**] Un'eccezione di tipo controllato viene catturata e gestita automaticamente dal garbage collector. [x] T [ ] F

10. [**Java**] Per rendere visibile il campo `p` di tipo `int` di una classe `C` è necessario dichiarare il campo come `package <nome_package> int p;` [ ] T [x] F

Prova di teoria 19 Luglio 2022

FONDAMENTI DI PROGRAMMAZIONE B

Tempo a disposizione: 30 minuti

Nome ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. [C++] Si considerino le classi `A`, `B` e `C`. La classe `B` è derivata da `A`, mentre la classe `C` è derivata da `B`. La seguente funzione

```
int move(B r) {...}
```

può accettare come argomenti oggetti

[a] esclusivamente di tipo `B` e delle sue superclassi
[b] di tipo `A` e `C`
[c] ✓ di tipo `B` e `C`
[d] di tipo `A`, `B` e `Object` ma non `C`
[e] nessuna delle precedenti

2. [C++] Si consideri il seguente programma

```
void f() {
  throw "helloworld";
}

int main() {
  try {
    f();
  } catch(int x) {
    cout << "1" << endl;
  } catch(Razionale z) {
    cout << "2" << endl;
  } catch(float z) {
    cout << "3" << endl;
  }
  return 0;
}
```

Si indichi cosa viene stampato a video dalla funzione `main`.

[a] `1 2` [b] `3` [c] `1 2 3` [d] ✓ nessuna delle precedenti

3. [C++] Qualsiasi oggetto viene allocato staticamente. [T] [F]

4. [C++] Si supponga che la classe `C` contenga il metodo `void f() const {...}`. Il metodo `f` della classe `C` è un metodo costante. [T] [F]

5. [C++] Una classe `C` ha a disposizione il distruttore esclusivamente se è stato implementato un costruttore. [T] [F]

6. [**Java**] Si considerino le classi `A`, `B` e `C`. Le classi `C` e `B` sono derivate da `A`. La classe `A` definisce un metodo `fun` che la classe `B`, ridefinisce mentre la classe `C` **non** ridefinisce. Si consideri il seguente frammento di codice.

```
A f = new C();
((A) f).fun();
```

- [ ] *a* viene sollevata una `ClassCastException`
- [ ] *b* viene invocato il metodo `fun` definito nella classe `B`
- [ ] *c* viene rilevato un errore a tempo di compilazione
- [x] *d* viene invocato il metodo `fun` definito nella classe `A`
- [ ] *e* nessuna delle precedenti

7. [**Java**] I valori di tipo primitivo vengono passati ad un metodo:

- [ ] *a* per valore oppure per riferimento utilizzando le classi wrapper
- [ ] *b* eclusivamente per riferimento
- [ ] *c* per valore oppure per riferimento utilizzando i puntatori
- [x] *d* nessuna delle precedenti

8. [**Java**] Si considerino le classi `A`, `B` e `C`. La classe `B` è derivata da `A`, mentre la classe `C` è derivata da `B`. Le classi di appartenenza della classe `B` sono

- [ ] *a* `A` e `Object`
- [ ] *b* `A`, `B`, `C` e `Object`
- [x] *c* `A`, `B` e `Object`
- [ ] *d* `A`, `B`, `C`
- [ ] *e* nessuna delle precedenti

9. [**Java**] Il campo `int p;` della classe `C` è un campo con visibilità di package. [x] T [ ] F

10. [**Java**] L'eccezione `public A extends RuntimeException {...}` è un eccezione di tipo controllato. [ ] T [x] F

Prova di teoria 2 Settembre 2022

FONDAMENTI DI PROGRAMMAZIONE B

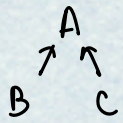
Tempo a disposizione: 30 minuti

Nome ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. [C++] Si considerino le classi `A`, `B` e `C`. Le classi `B` e `C` sono derivate da `A`. La seguente funzione

```
int move(B r) {...}
```



può accettare come argomenti oggetti

- [a] di tipo `A` e `B`
- [b] di tipo `B` e `C`
- [c] di tipo `A`, `B` e `C`
- [d] esclusivamente di tipo `B` e delle sue superclassi
- [e] esclusivamente di tipo `B` e delle sue sottoclassi

2. [C++] Se una classe definisce esplicitamente il distruttore, allora è opportuno

- [a] definire il costruttore senza parametri
- [b] definire esplicitamente anche il costruttore di copia e l'operatore di assegnamento
- [c] definire esplicitamente il costruttore di copia ma non l'operatore di assegnamento
- [d] nessuna delle precedenti

3. [C++] Si supponga che la classe `C` contenga il metodo `void f() const {...}`. Il metodo `f` della classe `C` è un metodo costante. [T] [F]

4. [C++] È possibile allocare gli oggetti esclusivamentre tramite allocazione dinamica [T] [F]

5. [C++] Si consideri il template di classe `Set<T>`. La classe `Set<int>` è derivata dalla classe `Set<float>` [T] [F]

6. [**Java**] Si considerino le classi `A`, `B` e `C`. La classe `B` è derivata da `A` e la classe `C` è derivata da `B`. La classe `B` definisce un metodo `g` che la classe `C` ridefinisce. Si consideri il seguente frammento di codice.

```
C o = new C();
((B) o).g();
```

[a] viene sollevata una `ClassCastException`
[b] viene invocato il metodo `g` definito nella classe `B`
[c] viene rilevato un errore a tempo di compilazione
[d] viene invocato il metodo `g` definito nella classe `C`
[e] nessuna delle precedenti

7. [**Java**] Si consideri la seguente dichiarazione di attributo all'interno di una classe `C`:

```
private final int x;
```

Si indichi la risposta corretta.

[a] è un attributo di classe con visibilità privata e modificabile
[b] è un attributo d'istanza con visibilità privata e non modificabile (dopo la sua inizializzazione nel costruttore)
[c] è un attributo di classe con visibilità privata e non modificabile (dopo la sua inizializzazione nel costruttore)
[d] è un attributo d'istanza con visibilità di package e modificabile
[e] nessuna delle precedenti

8. [**Java**] Si considerino le classi `A`, `B`, `C`, `D`. La classe `B` è derivata da `A`, la classe `C` è derivata da `B` e la classe `D` è derivata da `C`. Le **classi di appartenenza** della classe `B` sono

[a] `A`, `Object`
[b] `A`, `B`, `Object`
[c] `B`, `C`, `D`
[d] `B`, `C`, `D`, `Object`
[e] nessuna delle precedenti

9. [**Java**] Se non viene specificato nessun modificatore di visibilità, un campo ha visibilità di package. [T] [F]

10. [**Java**] L'eccezione `public A extends RuntimeException {...}` è una eccezione di tipo non controllato. [T] [F]

FONDAMENTI DI PROGRAMMAZIONE A

Tempo a disposizione: 30 minuti

Nome ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. Data la dichiarazione C++: `int x=3, y=2;` quali sono il tipo e il valore dell'espressione `(x + y)/2`?

- [ ] *a* `float`, 2.5
- [x] *b* `int`, 2
- [ ] *c* errore a tempo di compilazione
- [ ] *d* nessuna delle precedenti

2. Per quali valori di `x`, variabile di tipo `int`, l'espressione `x >= 0 && x < 10` risulta **vera**?

- [ ] *a* nessun valore
- [x] *b* qualsiasi valore compreso fra 0 (incluso) e 9 (incluso)
- [ ] *c* qualsiasi valore compreso fra 0 (incluso) e 10 (incluso)
- [ ] *d* qualsiasi valore minore di 10
- [ ] *e* nessuna delle precedenti

3. La seguente dichiarazione di variabile `char s[10] = {'\0'};` inizializza `s` alla stringa vuota. [x] V [ ] F

4. Supponendo che le espressioni vengano valutate da sinistra verso destra, cosa stampa a video il seguente frammento di codice?

```
int x = 0, y = 1;
bool b = --x != y - 1 || ++x == y--;
cout << b << "␣" << x << "␣" << y << endl;
```

- [ ] *a* `1 0 0`
- [ ] *b* `0 -1 0`
- [x] *c* `1 -1 1`
- [ ] *d* `0 0 0`

5. Cosa stampa il seguente programma?

```
int V[5] = {5,4,6,3,1};
int i = 1;
cout << V[i] + V[i + 2] + 1;
```

- [ ] *a* `10`
- [ ] *b* `11`
- [x] *c* `8`
- [ ] *d* `7`
- [ ] *e* nessuna delle precedenti

6. Si consideri la seguente funzione

```
int f(int& x) {
   return ++x;
}
```

Si indichi cosa stampa a video l'esecuzione delle seguenti istruzioni

```
int z = 3;
cout << f(z) + 3;
```

- [ ] *a* 6
- [ ] *b* 7
- [ ] *c* 5
- [ ] *d* nessuna delle precedenti

7. Cosa stampa il seguente programma?

```
int x = 5;
int& y = x;
x++;
y--;
cout << x << "␣" << y;
```

- [ ] *a* 6 4
- [ ] *b* 6 5
- [ ] *c* 5 4
- [ ] *d* 5 5
- [ ] *e* nessuna delle precedenti

8. Cosa stampa il seguente programma?

```
int x[5] = {2,7,8,1,3};
cout << *(x + 1);
```

- [ ] *a* 3
- [ ] *b* 7
- [ ] *c* 8
- [ ] *d* 6
- [ ] *e* nessuna delle precedenti

9. Cosa stampa il seguente frammento di codice?

```
int a = 6, b = 6;
int* p = &a;
int* r = &b;
*p = *r - 1;
cout << a << "␣" << b << endl;
```

- [ ] *a* 5 5
- [ ] *b* 6 6
- [ ] *c* 5 6
- [ ] *d* nessuna delle precedenti

```
int a = 1;
cout << a ; // 1
cout << &a; // 0x7fff
int * b = &a;
cout << b ; // 0x7fff
cout << *b; // 1
cout << &b; // 0x8888
(int *) c = b
cout << c ; // 0x8888
cout << *c; // 1
```

10. Si consideri la seguente funzione ricorsiva

```
int f(int x) {
   if (x == 0)
      return 0;
   else
      return f(x-1);
}
```

La chiamata a funzione `f(4)` ritorna `0`

- [ ] *V*
- [ ] F

FONDAMENTI DI PROGRAMMAZIONE A

Tempo a disposizione: 30 minuti

Nome ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. Si consideri la seguente tipo struct

```
struct Point {
   int x;
   int y;
}
```

e la seguente dichiarazione: `Point p = {1, 2}`. L'espressione corretta per accedere al campo `x` di `p` è

- [ ] *a* `p->x`
- [x] *b* `p.x`
- [ ] *c* `(*p).x`
- [ ] *d* nessuna delle precedenti

2. Per quali valori di `x`, variabile di tipo `int`, l'espressione `x >= 0 || x < 10` risulta **vera**?

- [ ] *a* nessun valore intero
- [x] *b* qualsiasi valore intero
- [ ] *c* esclusivamente qualsiasi valore compreso fra 0 (incluso) e 9 (incluso)
- [ ] *d* esclusivamente per qualsiasi valore minore di 10
- [ ] *e* nessuna delle precedenti

3. La seguente dichiarazione di variabile `char s[5] = {'a', 'b', '\0'};` inizializza `s` alla stringa `"ab"`. [x] V [ ] F

4. Supponendo che le espressioni vengano valutate da sinistra verso destra, cosa stampa a video il seguente frammento di codice?

```
int x = 2, y = 1;
bool b = (x++ == y + 1) || (++x != y--);
cout << b << "␣" << x << "␣" << y << endl;
```

- [ ] *a* `0 4 0`
- [ ] *b* `1 2 2`
- [x] *c* `1 3 1`
- [ ] *d* `1 2 1`

5. Cosa stampa il seguente programma?

```
for (int i = 0; i < 5; i++) {
   if (i == 2)
      break;
   cout << i;
}
```

- [ ] *a* `1 2`
- [ ] *b* `0 1 2`
- [ ] *c* `0 1 2 3 4`
- [x] *d* `0 1`
- [ ] *e* nessuna delle precedenti

6. Si consideri la seguente funzione

```
int f(int& x) {
   return ++x;
}
```

Si indichi cosa stampa a video l'esecuzione delle seguenti istruzioni

```
int z = 3;
f(z);
cout << z;
```

- [ ] *a* 3
- [ ] *b* 4
- [ ] *c* 5
- [ ] *d* nessuna delle precedenti

7. Cosa stampa il seguente programma?

```
int x = 7;
int& z = x;
x = x + 1;
z = z - 1;
cout << x << "␣" << z;
```

- [ ] *a* 8 6
- [ ] *b* 8 7
- [ ] *c* 7 7
- [ ] *d* 7 6
- [ ] *e* nessuna delle precedenti

8. Cosa stampa il seguente programma?

```
int x[5] = {2,7,8,1,3};
cout << *(x + 2);
```

- [ ] *a* 3
- [ ] *b* 7
- [ ] *c* 8
- [ ] *d* 6
- [ ] *e* nessuna delle precedenti

9. L'espressione corretta per allocare un array dinamico di 10 elementi interi è `new int[10];` [ ] V [ ] F

10. Si consideri la seguente funzione ricorsiva

```
int f(int x) {
   if (x == 0)
      return 0;
   else
      return f(x-1);
}
```

La chiamata a funzione `f(4)` ritorna `0` [ ] V [ ] F

FONDAMENTI DI PROGRAMMAZIONE A

Tempo a disposizione: 30 minuti

Nome ..... Cognome ..... Matricola .....

Per accede alla prova di programmazione è necessario rispondere correttamente ad almeno il 70% delle domande

1. Cosa stampa il seguente programma?

```
for (int j = 0; j < 3; j++) {
   if (j == 1) {
      continue;
   }
   cout << i;
}
```

- [ ] *a* `1 2`
- [ ] *b* `0 2`
- [ ] *c* `0 1 2 3`
- [ ] *d* `0 1 2`
- [x] *e* nessuna delle precedenti

2. Per quali valori di `x`, variabile di tipo `int`, l'espressione `x > 0 && x < 10` risulta **vera**?

- [ ] *a* nessun valore intero
- [ ] *b* qualsiasi valore intero
- [x] *c* esclusivamente i valori compresi fra 1 (incluso) e 9 (incluso)
- [ ] *d* esclusivamente i valori minori strettamente di 10
- [ ] *e* nessuna delle precedenti

3. Supponendo che le espressioni vengano valutate da sinistra verso destra, cosa stampa a video il seguente frammento di codice?

```
int x = 2,  y = 2;
bool b = (++x == y + 1) || (++x != y--);
cout << b << "␣" << x << "␣" << y << endl;
```

- [x] *a* `0 4 1`
- [ ] *b* `1 3 2`
- [ ] *c* `1 3 3`
- [ ] *d* `1 2 2`

4. In C++, il passaggio di parametri per riferimento è implementato esclusivamente tramite puntatori V [x] F

5. Si consideri il seguente tipo struct

```
struct MyStruct {
   int first;
   int second;
}
```

e la seguente dichiarazione: `MyStruct s = {1, 2}`. L'espressione corretta per accedere al campo `first` di `s` è

- [ ] *a* `(*s).first`
- [ ] *b* `s->first`

c `s[first]`
d `s.first`
e nessuna delle precedenti

6. Cosa stampa il seguente frammento di codice?

```
int arr[5] = {5,4,6,3,1};
int i = 1;
cout << arr[i + 2] + 1;
```

a 7
b 4
c 2
d nessuna delle precedenti

7. Cosa stampa il seguente frammento di codice?

```
int arr[3] = {2,7,8};
cout << *(arr + 2);
```

a 2
b 7
c 8
d nessuna delle precedenti

8. Cosa stampa il seguente frammento di codice?

```
int a = 1;
int& b = a;
a++;
b++;
cout << a << "␣" << b;
```

a 2 2
b 2 3
c 3 3
d 3 2
e nessuna delle precedenti

9. La seguente dichiarazione di variabile `char s[42] = {'\0'};` inizializza `s` alla stringa vuota. V F

10. Si consideri la seguente funzione ricorsiva

```
int fun(int p) {
   if (p == 0)
      return 0;
   else
      return fun(p - 1);
}
```

La chiamata a funzione `fun(-2)` ritorna `0` V F