C++ Quiz

C++ User Group Udine

27 maggio 2015

Storia del linguaggio

Domanda 1

Come si chiamava il linguaggio che Bjarne Stroustrup cominciò a sviluppare all'AT&T nel 1979, e che venne successivamente chiamato C++?

1. C = C + 1
2. C only uglier
3. C with Classes

Domanda 1

Come si chiamava il linguaggio che Bjarne Stroustrup cominciò a sviluppare all'AT&T nel 1979, e che venne successivamente chiamato C++?

1. C = C + 1
2. C only uglier
3. C with Classes

Risposta corretta: C with Classes

Domanda 2

Il C++ è notoriamente un'estensione del linguaggio C.
Su che versione dello standard ISO C è basato lo standard C++11?

1. ISO C 89
2. ISO C 99
3. ISO C 11

Domanda 2

Il C++ è notoriamente un'estensione del linguaggio C.
Su che versione dello standard ISO C è basato lo standard C++11?

1. ISO C 89
2. ISO C 99
3. ISO C 11

Risposta corretta: ISO C 99
Il C++98 era basato sul C89, mentre gli standard C++11 e C++14 sono basati sullo standard ISO/IEC 9899:1999, aka C99

Classi, oggetti e valori

Domanda 3

Cosa stampa questo pezzo di codice C++?

```
class Base {
public:
          void a() { std::cout << "base a\n"; }
  virtual void b() { std::cout << "base b\n"; }
};
class Derived : public Base {
public:
  void a() { std::cout << "derived a\n"; }
  void b() { std::cout << "derived b\n"; }
};

Base *obj = new Derived;
obj->a();
obj->b();
```

Domanda 3

Risposta:

```
base a
derived b
```

Il comportamento polimorfico tipico della programmazione orientata agli oggetti si ottiene solo su funzioni membro contrassegnate con `virtual`

Domanda 4

```
struct Base {
  Base()  {
    std::cout << 1;
  }
  ~Base() {
    std::cout << 4;
  }
};
struct Derived : Base {
  Derived()  {
    std::cout << 2;
  }
  ~Derived() {
    std::cout << 3;
  }
};
```

Cosa stampa questo pezzo di codice C++?

```
Base *b = new Derived;
delete b;
```

1. 1 2 3 4
2. 1 2 4
3. 1 2 4 3

Domanda 4

```
struct Base {
  Base()  {
    std::cout << 1;
  }
  ~Base() {
    std::cout << 4;
  }
};
struct Derived : Base {
  Derived()  {
    std::cout << 2;
  }
  ~Derived() {
    std::cout << 3;
  }
};
```

Cosa stampa questo pezzo di codice C++?

```
Base *b = new Derived;
delete b;
```

1. 1 2 3 4
2. 1 2 4
3. 1 2 4 3

Risposta corretta: 1 2 4
Il distruttore della classe base deve essere `virtual` per assicurarsi che i distruttori delle classi derivate vengano chiamati correttamente anche in contesti polimorfici.

Domanda 5

Che differenza c'è tra questi due pezzi di codice?

```
int x;

std::cout << x;
```

```
int x{};

std::cout << x;
```

1. Sono equivalenti, e stampano entrambi zero
2. Sono equivalenti, ed entrambi causano undefined behaviour
3. Il primo causa undefined behaviour, il secondo stampa zero

Domanda 5

Che differenza c'è tra questi due pezzi di codice?

```
int x;

std::cout << x;
```

```
int x{};

std::cout << x;
```

1. Sono equivalenti, e stampano entrambi zero
2. Sono equivalenti, ed entrambi causano undefined behaviour
3. Il primo causa undefined behaviour, il secondo stampa zero

Risposta corretta: la n°3.
La variabile x nel primo esempio resta *non inizializzata*, e l'accesso causa undefined behaviour, mentre nel secondo subisce una *value-initialization* e viene inizializzata a zero

Domanda 6

Cosa stampa questa riga di codice C++?

```
unsigned int n = 25;
std::cout << (n - 26);
```

1. 0
2. UINTMAX
3. -1
4. Provoca undefined behaviour

Domanda 6

Cosa stampa questa riga di codice C++?

```
unsigned int n = 25;
std::cout << (n - 26);
```

1. 0
2. UINTMAX
3. -1
4. Provoca undefined behaviour

Risposta corretta: UINTMAX (solitamente $2^{32} - 1$)

Il literal "26" è un intero con segno, ma per l'operazione binaria viene convertito ad un intero senza segno, e il risultato teoricamente negativo viene rappresentato come un grande numero positivo.

Sintassi e grammatica

Domanda 7

Qual è l'output di questo programma C++?

```
std::cout << 2["C++"];
```

1. Questo è ostrogoto
2. "C++C++"
3. "+"

Domanda 7

Qual è l'output di questo programma C++?

```
std::cout << 2["C++"];
```

1. Questo è ostrogoto
2. "C++C++"
3. "+"

Risposta corretta: il programma stampa "+".
L'espressione `a[b]`, quando sono in gioco tipi primitivi, è per definizione uguale a `*(a + b)`. L'addizione è commutativa, quindi `2["C++"]` è uguale a `"C++"[2]`, che estrae il secondo elemento dalla stringa (che è di tipo `char[4]`).

Domanda 8

Qual è l'output di questo programma C++?

```
struct X {
  X(int x = 42) { std::cout << x << "\n"; }
};

int main() {
  X x(0);
  X y();

  return 0;
}
```

Domanda 8

Qual è l'output di questo programma C++?

```
struct X {
  X(int x = 42) { std::cout << x << "\n"; }
};

int main() {
  X x(0);
  X y();

  return 0;
}
```

Risposta corretta: Stampa solo uno "0".
La riga `X y();` non istanzia un oggetto, ma dichiara il prototipo di una funzione. Questa particolarità della grammatica viene spesso chiamata "The Most Vexing Parse"

Domanda 9

```
struct Base {
  void f() {
    std::cout << 42;
  }
};
struct Derived : Base {
  void f() {
    Base:f();
  }
};

Derived d;
d.f();
```

Qual è l'output di questo programma C++?

1. Non compila
2. Stampa "42"
3. Va in crash (o in loop)

Domanda 9

```
struct Base {
  void f() {
    std::cout << 42;
  }
};
struct Derived : Base {
  void f() {
    Base:f();
  }
};

Derived d;
d.f();
```

Qual è l'output di questo programma C++?

1. Non compila
2. Stampa "42"
3. Va in crash (o in loop)

Risposta corretta: la n°3

Nella riga `Base:f();` manca un "due punti". `Base:` è quindi una label (di quelle usate per l'istruzione `goto`). La funzione `f()` sta quindi richiamando infinitamente se stessa.

Domanda 10

Cosa stampa questo pezzo di codice C++?

```
double pi;

pi = 3,141592653589793;

std::cout << pi;
```

Domanda 10

Cosa stampa questo pezzo di codice C++?

```
double pi;

pi = 3,141592653589793;

std::cout << pi;
```

Risposta: stampa l'intero "141592653589793"

Il separatore decimale è il punto, non la virgola.
La virgola in C e C++ è un operatore binario che valuta entrambi i suoi argomenti e restituisce il valore dell'operando di destra.