Variabili

Approfondimento

Roberto Borelli

Maggio 2021

Obbiettivi:

1. Chiarire il conecetto di parametro.
2. Introdurre il concetto di varaibile globale, locale e scope.

Un approfondimento su questi aspetti, migliorerà la comprensione generale di come funziona un programma e aiuterà a capire meglio ricorsione e funzioni.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: definzione

- Sono variabili.
- Sono dichiarate prima del main e fuori da qualsiasi funzione.
- Sono accessibili da qualunque funzione.
  - In particolare nel main.
  - E in qualsiasi altra funzione.
- Se in una funzione modifico il valore di una variabile globale, la modifica sarà tangibile a tutti i punti in cui la variabile è riferita.

Variabili globali: esempio base

Esempio (Variabile x gloabale)

```
1
2  int x = 10;
3
4  void laMiaFunzione(){
5          ...
6  }
7
8  int main(){
9          ...
10 }
```

La variabile *x* è globale: è definita fuori dal *main* e fuori dalla funzione *laMiaFunzione*.

Variabili globali: esempio

```
1
2  int x = 10;
3
4  void laMiaFunzione(){
5          x = x * 2;
6  }
7
8  int main(){
9          x = 50;
10         laMiaFunzione();
11         cout<<x;
12 }
```

- x è dichiarata globale.
- che valore verrà stampato?

Variabili globali: esempio

```
1
2  int x = 10;
3
4  void laMiaFunzione(){
5          x = x * 2;
6  }
7
8  int main(){
9          x = 50;
10         laMiaFunzione();
11         cout<<x;
12 }
```

- x è dichiarata globale.
- che valore verrà stampato?

Variabili globali: esempio

```
1
2  int x = 10;
3
4  void laMiaFunzione(){
5          x = x * 2;
6  }
7
8  int main(){
9          x = 50;
10         laMiaFunzione();
11         cout<<x;
12 }
```

- x è dichiarata globale.
- che valore verrà stampato?

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili globali: esempio

```
1  int x = 10;
2
3  void laMiaFunzione(){
4    x = x * 2;
5  }
6
7  int main(){
8    x = 50;
9    laMiaFunzione();
10   cout<<x;
11 }
```

1 *X* viene dichiarata globale con valore iniziale 10.

7 Inizia il programma.

8 *X* prende valore 50.

9 Viene richiamata *laMiaFunzione*.

4 *X* prende valore 100.

10 Viene stampato 100.

Variabili locali: definizione

- Sono variabili.
- Sono dichiarate in una funzione (tra le due graffe).
  - In particolare possono essere dichiarate nel main.
  - E in qualsiasi altra funzione.
- Le variabili locali esistono **solo** nella funzione in cui sono state dichiarate.

Variabili locali: definizione

- Sono variabili.
- Sono dichiarate in una funzione (tra le due graffe).
  - In particolare possono essere dichiarate nel main.
  - E in qualsiasi altra funzione.
- Le variabili locali esistono **solo** nella funzione in cui sono state dichiarate.

Variabili locali: definizione

- Sono variabili.
- Sono dichiarate in una funzione (tra le due graffe).
  - In particolare possono essere dichiarate nel main.
  - E in qualsiasi altra funzione.
- Le variabili locali esistono **solo** nella funzione in cui sono state dichiarate.

Variabili locali: definizione

- Sono variabili.
- Sono dichiarate in una funzione (tra le due graffe).
  - In particolare possono essere dichiarate nel main.
  - E in qualsiasi altra funzione.
- Le variabili locali esistono **solo** nella funzione in cui sono state dichiarate.

Variabili locali: definizione

- Sono variabili.
- Sono dichiarate in una funzione (tra le due graffe).
  - In particolare possono essere dichiarate nel main.
  - E in qualsiasi altra funzione.
- Le variabili locali esistono **solo** nella funzione in cui sono state dichiarate.

Variabili locali: esempio base

```
1  void laMiaFunzione(){
2          x = x * 2;
3  }
4
5  int main(){
6          int x = 10
7          laMiaFunzione();
8  }
```

- x è dichiarata locale all'interno del main.
- x è visibile sono nel main.
- Errore di compilazione: non esiste x in laMiaFunzione.

Variabili locali: esempio base

```
1        void laMiaFunzione(){
2                x = x * 2;
3        }
4
5        int main(){
6                int x = 10
7                laMiaFunzione();
8        }
```

- x è dichiarata locale all'interno del main.
- x è visibile sono nel main.
- Errore di compilazione: non esiste x in laMiaFunzione.

Variabili locali: esempio base

```
1        void laMiaFunzione(){
2                x = x * 2;
3        }
4
5        int main(){
6                int x = 10
7                laMiaFunzione();
8        }
```

- x è dichiarata locale all'interno del main.
- x è visibile sono nel main.
- Errore di compilazione: non esiste x in laMiaFunzione.

Variabili locali: esempio base

```
1        void laMiaFunzione(){
2                x = x * 2;
3        }
4
5        int main(){
6                int x = 10
7                laMiaFunzione();
8        }
```

- x è dichiarata locale all'interno del main.
- x è visibile sono nel main.
- Errore di compilazione: non esiste x in laMiaFunzione.

Variabili locali: esempio

```
1 void laMiaFunzione(){
2         int x = x * 2;
3 }
4
5 int main(){
6         int x = 50;
7         laMiaFunzione();
8         cout<<x;
9 }
```

- x è dichiarata locale in main.
- x è dichiarata locale in laMiaFunzione.
- La x del main è diversa dalla x di laMiaFunzione.
- che valore verrà stampato?

Variabili locali: esempio

```
1 void laMiaFunzione(){
2         int x = x * 2;
3 }
4
5 int main(){
6         int x = 50;
7         laMiaFunzione();
8         cout<<x;
9 }
```

- x è dichiarata locale in main.
- x è dichiarata locale in laMiaFunzione.
- La x del main è diversa dalla x di laMiaFunzione.
- che valore verrà stampato?

Variabili locali: esempio

```
1 void laMiaFunzione(){
2         int x = x * 2;
3 }
4
5 int main(){
6         int x = 50;
7         laMiaFunzione();
8         cout<<x;
9 }
```

- x è dichiarata locale in main.
- x è dichiarata locale in laMiaFunzione.
- La x del main è diversa dalla x di laMiaFunzione.
- che valore verrà stampato?

Variabili locali: esempio

```
1 void laMiaFunzione(){
2         int x = x * 2;
3 }
4
5 int main(){
6         int x = 50;
7         laMiaFunzione();
8         cout<<x;
9 }
```

- x è dichiarata locale in main.
- x è dichiarata locale in laMiaFunzione.
- La x del main è diversa dalla x di laMiaFunzione.
- che valore verrà stampato?

Variabili locali: esempio

```
1 void laMiaFunzione(){
2         int x = x * 2;
3 }
4
5 int main(){
6         int x = 50;
7         laMiaFunzione();
8         cout<<x;
9 }
```

- x è dichiarata locale in main.
- x è dichiarata locale in laMiaFunzione.
- La x del main è diversa dalla x di laMiaFunzione.
- che valore verrà stampato?

Scope

- Tutto si si riassume al concetto di scope.

Scope

Lo scope (contesto, visibilità) di una variabile, è la regione del programma in cui la variabile può essere usata.

- In scope diversi possono esistere variabili diverse ma con lo stesso nome.
- Variabili locali possono essere viste solo nello scope in cui sono dichiarate.
- Variabili globali possono essere viste da tutto il programma.

Scope

- Tutto si si riassume al concetto di scope.

Scope

Lo scope (contesto, visibilità) di una variabile, è la regione del programma in cui la variabile può essere usata.

- In scope diversi possono esistere variabili diverse ma con lo stesso nome.
- Variabili locali possono essere viste solo nello scope in cui sono dichiarate.
- Variabili globali possono essere viste da tutto il programma.

Scope

- Tutto si si riassume al concetto di scope.

Scope

Lo scope (contesto, visibilità) di una variabile, è la regione del programma in cui la variabile può essere usata.

- In scope diversi possono esistere variabili diverse ma con lo stesso nome.
- Variabili locali possono essere viste solo nello scope in cui sono dichiarate.
- Variabili globali possono essere viste da tutto il programma.

Scope

- Tutto si si riassume al concetto di scope.

Scope

Lo scope (contesto, visibilità) di una variabile, è la regione del programma in cui la variabile può essere usata.

- In scope diversi possono esistere variabili diverse ma con lo stesso nome.
- Variabili locali possono essere viste solo nello scope in cui sono dichiarate.
- Variabili globali possono essere viste da tutto il programma.

Buone pratiche

- Ove possibile si preferisce dichiarare:
  - Costanti $\rightarrow$ globali
  - Variabili $\rightarrow$ locali
- Se le variabili sono locali possono essere modificate solo nel loro scope.
- Quindi se sono in cerca di un errore riguardo a un valore della variabile x, l'errore può essere 'solo' nello scope della variabile x.
- Se x fosse globale, l'errore potrebbe essere ovunque. Assurdo!

Buone pratiche

- Ove possibile si preferisce dichiarare:
  - Costanti $\rightarrow$ globali
  - Variabili $\rightarrow$ locali
- Se le variabili sono locali possono essere modificate solo nel loro scope.
- Quindi se sono in cerca di un errore riguardo a un valore della variabile x, l'errore può essere 'solo' nello scope della variabile x.
- Se x fosse globale, l'errore potrebbe essere ovunque. Assurdo!

Buone pratiche

- Ove possibile si preferisce dichiarare:
  - Costanti $\rightarrow$ globali
  - Variabili $\rightarrow$ locali
- Se le variabili sono locali possono essere modificate solo nel loro scope.
- Quindi se sono in cerca di un errore riguardo a un valore della variabile x, l'errore può essere 'solo' nello scope della variabile x.
- Se x fosse globale, l'errore potrebbe essere ovunque. Assurdo!

Buone pratiche

- Ove possibile si preferisce dichiarare:
  - Costanti $\rightarrow$ globali
  - Variabili $\rightarrow$ locali
- Se le variabili sono locali possono essere modificate solo nel loro scope.
- Quindi se sono in cerca di un errore riguardo a un valore della variabile x, l'errore può essere 'solo' nello scope della variabile x.
- Se x fosse globale, l'errore potrebbe essere ovunque. Assurdo!

Buone pratiche

- Ove possibile si preferisce dichiarare:
  - Costanti $\rightarrow$ globali
  - Variabili $\rightarrow$ locali
- Se le variabili sono locali possono essere modificate solo nel loro scope.
- Quindi se sono in cerca di un errore riguardo a un valore della variabile x, l'errore può essere 'solo' nello scope della variabile x.
- Se x fosse globale, l'errore potrebbe essere ovunque. Assurdo!