Espressioni numeriche e binarie, conversione, selezione binaria

Docente: Dragan Ahmetovic
mail: dragan.ahmetovic@unimi.it
Ricevimento: su appuntamento

Tutor: Alexandru David

Sito del corso: `http://dragan.ahmetovic.it/?p=teaching`

Informazioni sul corso (cont.)

Sito consegna

- Potete caricare gli esercizi che fate (così controllo se sta andando bene)
  `https://upload.di.unimi.it/`
- Cercate di caricare gli esercizi fatti in classe e a casa
  (sopratutto se non siete sicuri di qualcosa)
- Se ci sono questioni o domande specifiche mettetemele come commenti

Questionario sulla volta scorsa

`dragan.ahmetovic.it/r.php`
(indirizzo anche sul sito)

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!
- Quando si scrive da tastiera lo spazio ed invio dividono le variabili

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!
- Quando si scrive da tastiera lo spazio ed invio dividono le variabili
- Ci sono altre varianti (es: ScanIn - solo spazio divide le variabili, invio termina)

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!
- Quando si scrive da tastiera lo spazio ed invio dividono le variabili
- Ci sono altre varianti (es: ScanIn - solo spazio divide le variabili, invio termina)
- **ES**: Fate un programma in cui due interi vengono letti con Scan e scritti a schermo
  Nota: è bene fare un PrintIn dicendo cosa l'utente deve fare se no non si capisce

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!
- Quando si scrive da tastiera lo spazio ed invio dividono le variabili
- Ci sono altre varianti (es: ScanIn - solo spazio divide le variabili, invio termina)
- **ES**: Fate un programma in cui due interi vengono letti con Scan e scritti a schermo
  Nota: è bene fare un PrintIn dicendo cosa l'utente deve fare se no non si capisce
- Cosa succede se metto una lettera? e float?

Inserimento valori da standard input

Scan

È una funzione di **fmt** che serve per leggere da tastiera.

- Come si usa?
- Ditemelo voi!

```
fmt.Scan(&qui, &quo, &qua) //variabili, uno può metterne quante vuole
```

- Prende il contenuto scritto a tastiera e lo mette nelle variabili puntate (&nomevar "punta" a nomevar, serve alle funzioni per poter modificare le variabili)
- Le variabili specificate devono esistere!
- Quando si scrive da tastiera lo spazio ed invio dividono le variabili
- Ci sono altre varianti (es: ScanIn - solo spazio divide le variabili, invio termina)
- **ES**: Fate un programma in cui due interi vengono letti con Scan e scritti a schermo
  Nota: è bene fare un PrintIn dicendo cosa l'utente deve fare se no non si capisce
- Cosa succede se metto una lettera? e float?
- E se uso ScanIn e invio dopo un solo numero?

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.
- Nel caso di int, esiste anche l'operatore "resto" (%), che da il resto di una divisione

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.
- Nel caso di int, esiste anche l'operatore "resto" (%), che da il resto di una divisione

```
fmt.Println("44 gatti in fila per 6 col resto di", 44%6)
```

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.
- Nel caso di int, esiste anche l'operatore "resto" (%), che da il resto di una divisione

```
fmt.Println("44 gatti in fila per 6 col resto di", 44%6)
```

- Operazioni brevi (per +,-,*,/,%): `i+=1 //uguale a i=i+1`

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.
- Nel caso di int, esiste anche l'operatore "resto" (%), che da il resto di una divisione

```
fmt.Println("44 gatti in fila per 6 col resto di", 44%6)
```

- Operazioni brevi (per +,-,*,/,%): `i+=1 //uguale a i=i+1`
- Incrementer: `i++ //uguale a i=i+1` Decrementer: `i-- //uguale a i=i-1`

Operazioni

- Solite operazioni aritmetiche +,-,*,/ (altre: `https://golang.org/ref/spec#Arithmetic_operators`)
- **ATTENZIONE**, il risultato ha lo stesso tipo di var usate (devono corrispondere)
- **ATTENZIONE**, ci può essere perdita di informazioni (es: divisione tra int)
- **ES**: dichiarate tre coppie di variabili (int, float32, float64), con i valori 1 e 3. Dividete la prima per la seconda, salvate i risultati in c1, c2, c3 rispettivamente e stampateli.
- Nel caso di int, esiste anche l'operatore "resto" (%), che da il resto di una divisione

```
fmt.Println("44 gatti in fila per 6 col resto di", 44%6)
```

- Operazioni brevi (per +,-,*,/,%): `i+=1 //uguale a i=i+1`
- Incrementer: `i++ //uguale a i=i+1` Decrementer: `i-- //uguale a i=i-1`
- La libreria di sistema math ha costanti e funzioni matematiche utili (Pi, Pow, Sqrt ...)

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

```
float64(variabileFloat32)
```

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

```
float64(variabileFloat32)
```

- Mi serve per fare le operazioni fra variabili di tipo diverso

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

```
float64(variabileFloat32)
```

- Mi serve per fare le operazioni fra variabili di tipo diverso
- **ES**: nell'esercizio precedente, stampate la differenza tra c3 e c2

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

```
float64(variabileFloat32)
```

- Mi serve per fare le operazioni fra variabili di tipo diverso
- **ES**: nell'esercizio precedente, stampate la differenza tra c3 e c2
- **ATTENZIONE**, ci può essere perdita di informazioni (es: se uso un tipo meno preciso)

Conversioni

- Posso convertire da un tipo di variabile ad un altro. sintassi: nuovotipo(variabile) es:

```
float64(variabileFloat32)
```

- Mi serve per fare le operazioni fra variabili di tipo diverso
- **ES**: nell'esercizio precedente, stampate la differenza tra c3 e c2
- **ATTENZIONE**, ci può essere perdita di informazioni (es: se uso un tipo meno preciso)
- **ES**: definite un int64 col valore 10000000000 (10 zeri), convertitelo in int 32

Operazioni Booleane

- Il tipo bool definisce due valori: true o false

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

- Operazioni tra i bool (logical operators):

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

- Operazioni tra i bool (logical operators):

```
true && false //true and false = false
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

- Operazioni tra i bool (logical operators):

```
true && false //true and false = false
true || false //true or true = true
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

- Operazioni tra i bool (logical operators):

```
true && false //true and false = false
true || false //true or true = true
!true //not true = False
```

Operazioni Booleane

- Il tipo bool definisce due valori: true o false
- Operazioni che restituiscono un bool (comparison operators):

```
1 == 2 //false
"pippo" != "pluto" //true
1.1 > 2.1 //false
2 <= 2.0 //true
```

- Operazioni tra i bool (logical operators):

```
true && false //true and false = false
true || false //true or true = true
!true //not true = False
```

- **ES**: dati due numeri in input restituite true se il primo è più grande e false altrimenti

if

servono per far fare cose diverse al programma nel caso qualcosa sia vero o falso

- sintassi:

```
if a == 5 { //if <valore booleano>
    fmt.Println("a è 5") //fai qualcosa
}
```

- ATTENZIONE: variabili dichiarate dentro al if sono valide solo dentro al if

if...else

- sintassi:

```
if a == 5 { //if <valore booleano>
    fmt.Println("a è 5") //fai qualcosa
} else {
    fmt.Println("a non è 5") //fai qualcosa altro
}
```

if... else if

- sintassi:

```
if a == 5 { //if <valore booleano>
    fmt.Println("a è 5") //fai qualcosa
} else if a == 4 {
    fmt.Println("a è 4") //fai qualcosa altro
}
```

if... else if ... else

- sintassi:

```
if a == 5 { //if <valore booleano>
    fmt.Println("a è 5") //fai qualcosa
} else if a == 4 {
    fmt.Println("a è 4") //fai qualcosa altro
} else {
    fmt.Println("a non è ne 5 ne 4") //fai altro ancora
}
```

0. comprensione del codice

Cosa restituisce questo codice per x = 75?

```
if(x > 25) {
println(x + " is greater than 25! " );
} else if (x > 50) {
println(x + " is greater than 50! " );
} else {
println(x + " is 25 or less! " );
}
```

1. inch_feet_yard_mile.go

Create un programma che, dato in input (Scan) un numero di inch, restituisca la sua conversione in miglia, yard, feet e inch

2. even_odd.go

Create un programma che, dato in input un numero intero, restituisca "pari" o "dispari" a seconda se il numero è pari o dispari

3. user_pass.go

Create un programma che chieda un nome utente ed una password e restituisca:

- "granted" se corretti
- "user wrong" se username errato
- "pass wrong" **altrimenti**

4. vowel_consonant.go

Create un programma che chieda una stringa e restituisca:

- "vowel" se inizia per vocale
- "consonant" se inizia per consonante

Hint: per accedere alla prima lettera di una stringa usate `nomeStringa[:1]`

5. quadrant.go

Create un programma che, date le coordinate (x e y) di un punto, dica se il punto è nel primo, secondo, terzo o quarto quadrante del piano cartesiano

6. triangle.go

Create un programma che, date le lunghezze di 3 segmenti, dica se si può generare un triangolo

7. how_much.go

Create un programma che calcoli il costo di un ordine di matite (numero dato dall'utente) sapendo che:

- il costo di una matita è 3 euro
- per ordini superiori a 50 matite c'è uno sconto del 10%
- per ordini superiori a 100 matite c'è uno sconto del 20%
- c'è un costo fisso di spedizione di 8 euro
- c'è la spedizione gratuita col codice sconto "hakunamatita"

8. happy_days.go

Create un programma che stampi "Happy Days" se oggi è domenica o lunedì

Esercizi

Ulteriori esercizi

- https://gitlab.di.unimi.it/laboratorio-di-programmazione/LabPubblico/tree/master/bianchessi_casazza/Laboratorio_2/
- https://homes.di.unimi.it/capra/labprog1920/lezioni/0011/