Strutture di controllo del flusso di esecuzione

Abbiamo diversi costrutti per regolare il flusso di un programma:

- IF – ELSE
- WHILE e DO-WHILE
- FOR e FOREACH
- SWITCH - CASE

```
 if ( condizione )
    blocco di istruzioni

 if ( condizione ) {
    ...istruzioni...
    }

 if ( vendite >= target ) {
    bonus = 100;
}
```

Costrutto IF

- E' un'istruzione condizionale, permette cioè di eseguire un blocco di istruzioni solo se si verifica una determinata condizione.

```
//SINTASSI:
 if ( condizione )
    blocco di istruzioni

//CONDIZIONE BOOLEANA CON OPERATORI RELAZIONALI
 if ( condizione ) {
    ...istruzioni...
    }

//ESEMPIO:
 if ( vendite >= target ) {
    bonus = 100;
}
```

Costrutto IF - ELSE

- Indichiamo anche cosa fare se non si supera la condizione

```
//SINTASSI:
if ( condizione )
    //istruzioni se supera la condizione
else
    //istruzioni se NON supera la condizione

if ( condizione ) {
    ...istruzioni...
}
else {
    ...istruzioni...
}

//ESEMPIO:
 if ( vendite >= target ) {
bonus = 100; }
else
{
bonus = -100;
}
```

WHILE: Ciclo Condizionato

- Nel caso del while si ha un ciclo condizionato: si ripete il ciclo fintanto che una condizione è verificata (è vera)
- LA CONDIZIONE BOOLEANA `true/false`
- DETERMINA LA CONTINUAZIONE DEL PROGRAMMA
- ED ESEGUE L'ELENCO DELLE OPERAZIONI DEL BLOCCO

WHILE: Ciclo Condizionato

- Si indica la condizione che deve essere vera per far ripetere il ciclo
- `while ( condizione )`
- `blocco di istruzioni da ripetere`

SINTASSI:

```
  while ( condizione ) {
...istruzioni da ripetere... }
```

ESEMPIO:

```
while ( ricavo < obiettivo )
{
ricavo++;
}
```

DO-WHILE: Ciclo Condizionato

- Si verifica la condizione dopo il primo ciclo SINTASSI:

```
do
while ( condizione )

   blocco di istruzioni da ripetere
```

```
 do {
...istruzioni da ripetere almeno una volta...
}
while ( condizione )
```

FOR: Ciclo Determinato

- Da usare quando è noto a priori il numero di volte che voglio ripetere un ciclo

```
for (partenza, fine, incremento)
blocco di istruzioni da ripetere
```

SINTASSI:

```
for( int i=0; i <= 10; i++ ) {
    ...istruzioni da ripetere...
}
```

ESEMPIO:

```
for (int i=0; i<5; i++) {
    System.out.println(“Passo: “ + i);
}
```

Costrutto SWITCH-CASE:

- In base al valore assunto da una variabile sceglie le istruzioni da eseguire
- se la variabile v vale...

1. CODICE DA ESEGUIRE
2. CODICE DA ESEGUIRE
3. CODICE DA ESEGUIRE

- ...altrimenti c'è un caso di `default`, ma occhio al `tipo` del dato!

Costrutto SWITCH-CASE:

SINTASSI:

```
switch(s)
{
case 1:
...istruzioni...
break;
case 2:
...istruzioni...
break;
default:
...istruzioni per input non corretto...
break;
}
```

Costrutto SWITCH-CASE:

ESEMPIO:

```
Scanner sca = new Scanner(System.in);
int s = sca.nextInt();
switch(s)
{
    case 1:
    System.out.prinln(“la s è: ” + s);
    break;
    case 2:
    System.out.prinln(“la s è: ” + s);
    break;
    default:
    System.out.prinln(“la s ha valore non supportato);
    break;
}
```

Interrompere un ciclo

BREAK SENZA ETICHETTA:

- Se stiamo eseguendo un ciclo, possiamo utilizzare la parola `break` per interromperlo in qualsiasi momento.

```
for (int i=0; i<=10; i++) {
. . . codice prima . . .
    if (condizione){
        break;
    }
. . . codice dopo . . .
}
```

- Si interrompe quindi il ciclo e si riprende l'esecuzione delle istruzioni al di fuori di esso.
- SE LA CONDIZIONE VIENE SUPERATA E SI ESEGUE IL **BREAK** ALLORA SI ESCE DAL FOR E NON LO SI CONTINUA PIÙ. NON VIENE ESEGUITO IL codice dopo E NON VENGONO NEANCHE ESEGUITI ALTRI CICLI DEL FOR SE VE NE ERANO RIMASTI

Interrompere un ciclo

BREAK CON ETICHETTA:

```
etichetta1:
while ( . . .) {
    for ( . . . ) {
    . ..
    if (condizione){
        break etichetta1;
        }
        . . .
    }
}
```

SE LA CONDIZIONE VIENE SUPERATA E SI ESEGUE IL `break etichetta1` SI INTERROMPE IL ciclo INDICATO DALL' `etichetta1`.

Interrompere un ciclo

CONTINUE: anzichè terminare completamente il ciclo ed uscire fuori, interrompe solo l'iterazione corrente e passa alla successiva.

```
for (int i=0; i<=10; i++) {
    . . . codice prima . . .

    if (condizione){

        continue;
    }

    . . . codice dopo . . .
}
```

- SE LA CONDIZIONE VIENE SUPERATA E SI ESEGUE IL CONTINUE ALLORA SI ESCE DALL'ITERAZIONE ATTUALE, NON VIENE ESEGUITO IL codice dopo E SI PASSA ALLA SUCCESSIVA ITERAZIONE DEL FOR.