# Control Statements

## ‘for’ Loop …

General form

```
for(initialization; condition; iteration) {
   //  body
}
```

example:

```
for( n=10; n>0; n--) {
  System.out.println(n);
}
```

## ‘for’ Loop (cont..)

- Usually variables that controls the ‘*for’* loop is needed only inside the loop block
  - Thus variable declaration may be done inside the *initialization* portion

```
for(int n=10; n>0; n--) {
  System.out.println(n);
}
```

- Scope of variable ends when ‘for’ loop finished execution

## Jump Statements

# ‘break’ to exit a loop

- Using ‘break’ to exit a loop
  - bypassing the conditional expression and remaining code

```
for (int i=0; i<10; i++) {
    if(i = =5 ) break;
    System.out.println(“i: ” + i);
}
```

Output

```
i:  0
i:  1
i:  2
i:  3
i:  4
```

# ‘break’ as GoTo

- Java doesn’t have ‘goto’ statement
  - but ‘break’ statement provide a ’civilized’ form of ‘goto’

    break *label;*

```
outer: for (int i=0; i<3; i++) {
        System.out.println(“i” + i);
        for (int j=0; j<100; j++) {
                if(j = = 10) break outer;   // exit both loops
                System.out.println(j + “ ”);
        }
}
```

- Output: 0 1 2 3 4 5 6 7 8 9

## ‘break’ as ‘goto’ (cont..)

- You can’t break any label which is not defined for an enclosing block

```
one: for(int i=0; i<3 ; i++) {
        System.out.println(i);
}
for(int j=0; j<100 ; j++) {
        if(j = = 10) break one;     // WRONG
        System.out.println(j);
}
```

## Continue

- Might want to continue running the loop, but stop processing remainder of the loop

```
for(int i=0; i<10; i++) {
        System.out.print(i+ " " );
        if (i % 2 = = 0) continue;
        System.out.println(" ");
}
```

Output

```
0  1
2  3
4  5
6  7
8  9
```

## ‘continue’ as ‘goto’

```
outer: for(int i=0; i<5; i++) {
            for(int j=0; j<5; j++) {
                  if( j > i) {
                      System.out.println( );
                      continue outer;
                  }
                  System.out.print ( i * j );
              }
         }
```

```
0
0  1
0  2  4
0  3  6  9
0  4  8  12  16
```