CS 241 — Introduction to Problem Solving and Programming

Fundamentals of Programming

Flow of control, part IV : Uses of loops (and other miscellany)

Feb 2, 2005

Overview

- A little new technical information
- Uses
- Warnings and debugging tips

Tech info: The switch statement

Recall the multibranch `if-else` statement.

```
if (guess == 16)
   System.out.println("That is correct!");
else if (guess < 1)
   System.out.println("That's not even in the range.");
else if (guess > 99)
   System.out.println("That's not even in the range.");
else
   System.out.println("I'm sorry, " + guess + " is wrong.");
```

Switch statement

This pattern occurs frequently, especially when checking integer equality.

```
String name = DocsIO.readString("Please enter your name--> ");
System.out.println("Please choose a preferred title: \n   " +
                   "\t1. Mr \n \t2. Miss\n \t3. Ms\n " +
                   "\t4. Mrs\n \t5. Dr\n \t6. Rev");
int titleCode = DocsIO.readint("Your choice--> ");

if (titleCode == 1)  name = "Mr. " + name;
else if (titleCode == 2)   name = "Miss " + name;
else if (titleCode == 3)   name = "Ms. " + name;
else if (titleCode == 4)   name = "Mrs. " + name;
else if (titleCode == 5)   name = "Dr. " + name;
else if (titleCode == 6)   name = "Rev. " + name;
else  System.out.println("Invalid choice; using none.");

System.out.println("Hello, " + name);
```

Switch statement

This pattern can be executed more efficiently using a switch statement:

```
switch(titleCode) {
case 1:
    name = "Mr. " + name;
    break;
case 2:
    name = "Miss " + name;
    break;
case 3:
    name = "Ms. " + name;
    break;
case 4:
    name = "Mrs. " + name;
    break;
case 5:
    name = "Dr. " + name;
    break;
case 6:
    name = "Rev. " + name;
    break;
default:
    System.out.println("Invalid choice; using none.");
}
```

Switch statement

Specification of the `switch` statement:

For statement:

```
switch (IntegralExpression) {
          case IntegralLiteral :
              Statement
              Statement
          ...
          default :
              Statement
              Statement
          ...
}
```

Switch statement

case turns flow of control on.

```
int input = DocsIO.readint("Please enter a number--> ");
switch(input) {
case 1:
    System.out.println("Aloha");
case 2:
    System.out.println("Salve");
    break;
case 3:
    System.out.println("Ahoy");
default:
    System.out.println("Hola");
}
```

```
ar1121: {36} java GoofySwitch
Please enter a number--> 1
Aloha
Salve
ar1121: {37} java GoofySwitch
Please enter a number--> 2
Salve
ar1121: {38} java GoofySwitch
Please enter a number--> 3
Ahoy
Hola
ar1121: {39} java GoofySwitch
Please enter a number--> 4
Hola
```

Switch statement

`default` is optional

```
int input = DocsIO.readint("Please enter a number--> ");
switch(input) {
case 1:
    System.out.println("Aloha");
case 2:
    System.out.println("Salve");
    break;
case 3:
    System.out.println("Ahoy");
}
```

```
ar1121: {52} java GoofySwitch
Please enter a number--> 1
Aloha
Salve
ar1121: {53} java GoofySwitch
Please enter a number--> 4
ar1121: {54} java GoofySwitch
Please enter a number--> 3
Ahoy
```

Switch statement

char counts as “integral.”

```
char input = DocsIO.readchar("Please enter a letter--> ");
switch(input) {
case 'A' : case 'a':
    System.out.println("Aloha");
case 'B': case 'b':
    System.out.println("Salve");
    break;
default:
    System.out.println("Ahoy");
}
```

```
ar1121: {61} java GoofySwitch
Please enter a letter--> a
Aloha
Salve
ar1121: {62} java GoofySwitch
Please enter a letter--> B
Salve
ar1121: {63} java GoofySwitch
Please enter a letter--> c
Ahoy
```

Tech info: Conditional operator

Another common pattern is

```
if (n1 > n2)
    max = n1;
else
    max = n2;
```

This can be expressed as

```
max = (n1 > n2) ? n1 : n2;
```

Conditional operator

? and : together make the conditional operator (or ternary operator).

BooleanExpression ? *Expression*–if-true : *Expression*–if-false;

Tech info: Commas in the headers

The first and last slots in the `for` header can contain several expressions, separated by commas.

```
int total = 0;
int number = DocsIO.readint("How many values do you want to average? " );
int current = 0;
while (current < number) {
    total += DocsIO.readint("Please enter next value--> ");
    current++;
}
System.out.println("Average: " + ((double) total / number));
```

Commas in the headers

The first and last slots in the `for` header can contain several expressions, separated by commas.

```
int total = 0;
int number = DocsIO.readint("How many values do you want to average? " );
for (int current = 0; current < number; current ++)
    total += DocsIO.readint("Please enter next value--> ");
System.out.println("Average: " + ((double) total / number));
```

Commas in the headers

The first and last slots in the `for` header can contain several expressions, separated by commas.

```
for (int total = 0,
     int number = DocsIO.readint("How many values do you want to average? " ),
     int current = 0;
     current < number;
     total += DocsIO.readint("Please enter next value--> "),
     current ++)
   ;
System.out.println("Average: " + ((double) total / number));
```

Tech info: Continue

The `continue` statement works like `break` except that it jumps to the beginning of the loop.

```
for (  ;  ; ) {
        . . .
        if ( . . . )
              break;            // quit the loop
        . . .
        if (. . .)
              continue;     // skip the rest of the loop and start over
        . . .
}
```

Continue

Revising out occurrence-counter example with a `continue`.

```
String text = DocsIO.readString("Please enter the string--> ");
char searchItem = DocsIO.readchar("Please enter the search item--> ");
int occurrences = 0;

for (int position = 0; position < text.length(); position++)
    char current = text.charAt(position);
    if (current != searchItem)
        continue;
    occurrences++;
    System.out.println("Found occurrence " + occurrences +
                       " at position " + position);
}
System.out.println(occurrences + " occurrences found.");
```

Continue

Revising out occurrence-counter example with a `continue`.

```
String text = DocsIO.readString("Please enter the string--> ");
char searchItem = DocsIO.readchar("Please enter the search item--> ");
int occurrences = 0;

for (int position = 0; position < text.length(); position++)
   char current = text.charAt(position);
    if (current == searchItem) {
        occurrences++;
        System.out.println("Found occurrence " + occurrences +
                           " at position " + position);
    }
}
System.out.println(occurrences + " occurrences found.");
```

Tech info: Exit

Sometimes you may want to force your program to quit—not just the loop, but the whole program.

Then use the `exit` method:

```
System.exit(0);
```

(Actually, other numbers besides zero would work, but zero sends a standard signal to the operating system.)

Exit

```
for(;;) {
   . . .
   System.out.println("4. Quit and show results");
   System.out.println("5. Quit immediately");
   . . .
   else if (query == 4)
       break;
   else if (query == 5)
       System.exit(0);
}
```

Uses: Standard pseudocode

The Fibonacci numbers are a sequence, beginning with 0 and 1, such that all subsequent numbers are equal to the sum of the two previous.

$$\begin{aligned}
f_1 &= & & & & & 0\\
f_2 &= & & & & & 1\\
f_3 &= & f_1 + f_2 &= & 0+1 &= & 1\\
f_4 &= & f_2 + f_3 &= & 1+1 &= & 2\\
f_5 &= & f_3 + f_4 &= & 1+2 &= & 3\\
f_6 &= & f_4 + f_5 &= & 2+3 &= & 5
\end{aligned}$$

What algorithm can we formulate for calculating the first 15 Fibonacci numbers?

Pseudocode

The standard way to express a counting loop in pseudocode is to write "for (variable) equals (initial value) to (terminal value)."

- Initialize `previous` to zero.
- Print `previous`
- Initialize `current` to one.
- Print `current`
- for `i` = 3 to 15
  - `next` =

Pseudocode

The standard way to express a counting loop in pseudocode is to write “for (variable) equals (initial value) to (terminal value).”

- Initialize `previous` to zero.
- Print `previous`
- Initialize `current` to one.
- Print `current`
- for `i` = 3 to 15
  - `next = current + previous`

Pseudocode

The standard way to express a counting loop in pseudocode is to write "for (variable) equals (initial value) to (terminal value)."

- Initialize `previous` to zero.
- Print `previous`
- Initialize `current` to one.
- Print `current`
- for `i` = 3 to 15
  - `next = current + previous`
  - `previous = current`

Pseudocode

The standard way to express a counting loop in pseudocode is to write "for (variable) equals (initial value) to (terminal value)."

- Initialize `previous` to zero.
- Print `previous`
- Initialize `current` to one.
- Print `current`
- for `i` = 3 to 15
  - `next` = `current` + `previous`
  - `previous` = `current`
  - `current` = `next`
  - Print `current`

Fibonacci numbers

```
int previous = 0;
System.out.println("F1: 0");
int current = 1;
System.out.println("F2: 1");

for (int i = 3; i < 16; i++) {
    int next = current + previous;
    previous = current;
    current = next;
    System.out.println("F" + i + ": " + current);
}
```

Fibonacci numbers

```
F1: 0
F2: 1
F3: 1
F4: 2
F5: 3
F6: 5
F7: 8
F8: 13
F9: 21
F10: 34
F11: 55
F12: 89
F13: 144
F14: 233
F15: 377
```

Fibonacci numbers

Make sure you can use both count-controlled loops and ask-before-iterating loops.

```
int previous = 0;
System.out.println("F1: 0");
int current = 1;
System.out.println("F2: 1");

for (;;) {
    char query = DocsIO.readchar("Another? (y/n)--> ");
    if (query == 'n' || query == 'N') break;
    int next = current + previous;
    previous = current;
    current = next;
    System.out.println("F" + i + ": " + current);
}
```

Uses: Menu-driven

Let's put several things together for a big example.

We want to calculate various formulas on circles and spheres, based on the radius, inputted from the user.

After inputting the radius once, repeatedly offer the user a choice of formulas, executed with a switch statement.

This style of program is menu-driven.

Menu-driven

```
public class Circle {
    public static void main(String[] args) {
        String menu =
            "\t1. Diameter of circle\n" +
            "\t2. Circumference of circle\n" +
            "\t3. Area of circle\n" +
            "\t4. Volume of sphere\n" +
            "\t5. Surface area of sphere\n" +
            "\t6. Quit\n";
        double radius = DocsIO.readdouble("Please enter the radius--> ");
```

Menu-driven

```
for (;;) {
    System.out.println(menu);
    int query = DocsIO.readint("Your choice-->");
    boolean runAgain = true;
    switch(query) {
    case 1:
        System.out.println("Diameter: " + (2 * radius));
        break;
    case 2:
        System.out.println("Circumference: " + (2 * radius * 3.14159));
        break;
    case 3:
        System.out.println("Area: " + (radius * radius * 3.14159));
        break;
    case 4:
        System.out.println("Volume: " +
                                   (radius * radius * radius * .75 * 3.14159));
        break;
```

Menu-driven

```
            case 5:
                System.out.println("Surface area: " +
                                   (4 * radius * radius * 3.14159));
                break;
            case 6:
                runAgain = false;
                break;
            default:
                System.out.println("Invalid choice, try again.");
            }
            if (!runAgain) break;
        }
        System.out.println("Thanks for using this program.");
    }
}
```

Menu-driven

```
ar1121: {86} java Circle
Please enter the radius--> 23.45
        1. Diameter of circle
        2. Circumference of circle
        3. Area of circle
        4. Volume of sphere
        5. Surface area of sphere
        6. Quit

Your choice-->1
Diameter: 46.9
        1. Diameter of circle
        2. Circumference of circle
        3. Area of circle
        4. Volume of sphere
        5. Surface area of sphere
        6. Quit
Your choice-->3
Area: 1727.5681949749996
        1. Diameter of circle
        2. Circumference of circle
        3. Area of circle
        4. Volume of sphere
        5. Surface area of sphere
        6. Quit

Your choice-->7
Invalid choice, try again.
        1. Diameter of circle
        2. Circumference of circle
        3. Area of circle
        4. Volume of sphere
        5. Surface area of sphere
        6. Quit

Your choice-->6
Thanks for using this program.
```

Uses: Nested loops

Recall our character counter example.

```
String text = DocsIO.readString("Please enter the string--> ");
char searchItem = DocsIO.readchar("Please enter the search item--> ");
int occurrences = 0;
int position = 0;

while(position < text.length()) {
    char current = text.charAt(position);
    if (current == searchItem) occurrences++;
    position++;
}
System.out.println(occurrences + " occurrences found.");
```

Nested loops

Suppose we want to find the occurrence of *every* character.

Assume one new `String` method:

StringExpression.`lastIndexOf`(*CharExpression*)

Returns the the position where the given character last occurs in the string, or -1 if it doesn't appear at all.

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {









}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {









    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;







    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;
        int occurrences = 0;






    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;
        int occurrences = 0;
        for (int j = i; j < text.length(); j++)




    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;
        int occurrences = 0;
        for (int j = i; j < text.length(); j++)
            if (searchItem == text.charAt(j))




    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;
        int occurrences = 0;
        for (int j = i; j < text.length(); j++)
            if (searchItem == text.charAt(j))
                occurrences++;


    }
}
```

Nested loops

```
String text = DocsIO.readString("Please enter the string--> ");
String usedLetters = "";

for (int i = 0; i < text.length(); i++) {
    char searchItem = text.charAt(i);
    if (usedLetters.lastIndexOf(searchItem) == -1) {
        usedLetters += searchItem;
        int occurrences = 0;
        for (int j = i; j < text.length(); j++)
            if (searchItem == text.charAt(j))
                occurrences++;
        System.out.println(occurrences +
                           " occurrences of " + searchItem);
    }
}
```

Nested loops

```
ar1121: {96} java SuperCharCounter
Please enter the string--> Arma virumque cano Troiae qui primus ob oris
1 occurrences of A
5 occurrences of r
3 occurrences of m
3 occurrences of a
7 occurrences of  
1 occurrences of v
5 occurrences of i
4 occurrences of u
2 occurrences of q
2 occurrences of e
1 occurrences of c
1 occurrences of n
4 occurrences of o
1 occurrences of T
1 occurrences of p
2 occurrences of s
1 occurrences of b
```

```
ar1121: {97} java SuperCharCounter
Please enter the string--> Eyore eeks eerily everywhere
1 occurrences of E
3 occurrences of y
1 occurrences of o
4 occurrences of r
9 occurrences of e
3 occurrences of  
1 occurrences of k
1 occurrences of s
1 occurrences of i
1 occurrences of l
1 occurrences of v
1 occurrences of w
1 occurrences of h
```

Tips: Multiple exit points

Our text has this example:

```
double total = 0;
for (int i = 0; i < 10; i++) {
    double amount =
        DocsIO.readdouble("Enter the cost of item #" + i + ": $");
    total += amount;
    if (total >= 100) {
        System.out.println("You spent all your money");
        break;
    }
    System.out.println("Your total so far is $" + total);
}
```

Multiple exit points

Loops with multiple exit points are hard to follow.

Avoid them, looking for equivalent loops with a single point of exit.

```
double total = 0;
for (int i = 0; i < 10  && total < 100; i++) {
    double amount =
        DocsIO.readdouble("Enter the cost of item #" + i + ": $");
    total += amount;
    System.out.println("Your total so far is $" + total);
}
```

Some might claim that using test-in-the-middle or `break` at all may not be the best programming practice.

Tips: Tracing loops

One important practice to help you think clearly about loops is to trace the variables. Try tracing the variables on this example.

```
int number = DocsIO.readint("Please enter a number to factor--> ");

int factor = 2;
while (number != 1) {
    if (number % factor == 0) {
        System.out.println(factor);
        number /= factor;
    }
    else
        factor++;
}
```

Tips: Tracing loops

To debug, add variable-tracing output.

```
int number = DocsIO.readint("Please enter a number--> ");

int factor = 2;
while (number != 1) {
    System.out.println("number: " + number);
    System.out.println("factor: " + factor);
    if (number % factor == 0) {
        System.out.println(factor);
        number /= factor;
    }
    else
        factor++;
}
```

```
Please enter a number--> 15
number: 15
factor: 2
number: 15
factor: 3
3
number: 5
factor: 3
number: 5
factor: 4
number: 5
factor: 5
5
```

Tips

What's wrong here?

```
String text = DocsIO.readString("Please enter the string--> ");

for (int i = text.length() - 1; i > 0; i--)
    System.out.println(text.charAt(i));
```

```
Please enter the string--> Wheaton
n
o
t
a
e
h
```

Tips: Off-by-one error

Beware the off-by-one error. Watch the endpoints.

```
        String text = DocsIO.readString("Please enter the string--> ");

        for (int i = text.length() - 1; i > 0; i--)
            System.out.println(text.charAt(i));
```

```
Please enter the string--> Wheaton
n
o
t
a
e
h
```

Tips: Infinite loops

Beware the infinite loop. Make sure the condition you're testing changes.

```
int number = DocsIO.readint("Please enter a number--> ");

int factor = 2;
while (number != 1) {
    System.out.println("number: " + number);
    System.out.println("factor: " + factor);
    if (number % factor == 0)
        System.out.println(factor);
    else
        factor++;
}
```

Summary

Be able to identify the following concepts

- Switch statement
- Conditional operator
- Continue statement
- Pseudocode: for i = 0 to n
- Menu-driven application
- Nested loops
- Off-by-one error

Also, be able to trace the execution of a loop by being able to discern the value of each variable at the beginning of each iteration.