Read: Sec. 2.3.8 and chapter 7 on loops. The lectures on the ProgramLive CD can be a big help.

Announcements

1. Prelim 2 next Tuesday evening, 7:30PM, Uris Auditorium.
   Yes, for-loops are not on this prelim.

2. Please complete an online questionnaire concerning your TA.
   http://www.engineering.cornell.edu/TAEval/menu.cfm
   This is a midterm evaluation. It is important, because your constructive comments are used to help the TA improve, which may help you in this course.
   You will probably receive an email about this. Please complete the survey this week!

Assertion: true-false statement placed in a program to assert that it is true at that place.

Precondition: assertion placed before a segment
Postcondition: assertion placed after a segment

Solving a problem

// x = sum of 1..n
x = x + n+1;
// x = sum of 1..n

What statement do you put here so that segment is correct? (if precondition is true, execution of segment should make postcondition true.)

Solving a problem

// x = sum of 1..n-1
x = x + n;
// x = sum of 1..n-1

What statement do you put here so that segment is correct? (if precondition is true, execution of segment should make postcondition true.)

Some anagrams

A decimal point I'm a dot in place
Debit card Bad credit
Dormitory Dirty room
Schoolmaster The classroom Slot machines
Statue of liberty Built to stay free Snooze alarms
The Morse code Here come dots
Western Union No wire unsept George Bush He bugs Gore
Parishioners I hire parsons

Circumstantial evidence Can ruin a selected victim
Victoria, England's queen Governs a nice quiet land
Eleven plus two Twelve plus one (and they have 13 letters?)

x = sum of 1..n
x ? n 1
x ? n 2
x ? n 0

x = sum of 1..n-1
n = n + 1;
// x = sum of 1..n-1

n n
1 2 3 4 5 6 7 8 9 10 11
x 15

x contains sum of these

n
1 2 3 4 5 6 7 8 9 10 11

x

15
Execution of the for-loop

The for-loop:

```
for (int i = 2; i <= 4; i = i + 1) {
    x = x + i * i;
}
```

1. **Loop counter**: i
2. **Initialization**: int i = 2;
3. **Loop condition**: i <= 4;
4. **Increment**: i = i + 1

### Iteration

1. Execution of repetend

To execute the for-loop:

1. Execute initialization.
2. If loop condition is false, terminate execution.
3. Execute the repetend.
4. Execute the increment and repeat from step 2.

```
i = 2;
```

```
i = i + 1;
```

The invariant is an assertion about the variables that is true before and after each iteration (execution of the repetend).

Methodology for developing a for-loop

1. Recognize that a range of integers b..c has to be processed
2. Write the command and equivalent postcondition.
3. Write the basic part of the for-loop.
4. Implement the repetend (Process k).
5. Figure out any initialization.
6. Implement the repetend (Process k).

```
for (int k = a; k <= b; k = k + 1) {
    Process k
}
```

// Postcondition: range b..c has been processed

// Process b..c
// Postcondition: range b..c has been processed

Finding an invariant: something that is true before and after each iteration (execution of the repetend).

```
for (int k = 0; k < s.length(); k = k + 1) {
    Process k;
}
```

// x = no. of adjacent equal pairs in s[0..s.length()-1]

What is the invariant?

A. x = no. adj. equal pairs in s[1..k]
B. x = no. adj. equal pairs in s[0..k]
C. x = no. adj. equal pairs in s[1..k–1]
D. x = no. adj. equal pairs in s[0..k–1]
E. None of these

An empty set of characters or integers has no maximum. Therefore, be sure that 0..k–1 is not empty. Therefore, start with k = 1.

Being careful

1. What is the invariant?
2. How do we initialize c and k?

A. k = 0; c = s.charAt[0];
B. k = 1; c = s.charAt[0];
C. k = 1; c = s.charAt[1];
D. k = 0; c = s.charAt[1];
E. None of the above