CS1110  22 October 2009
Read: Sec. 2.3.8 and chapter 7 on loops.
The lectures on the ProgramLive CD can be a big help.

Some anagrams
A decimal point
Tin a dot in place
Animosity
Is no amity
Debit card
Bad credit
Desperation
A rope ends it
Domino
Dirty mound
Funeral
Real fun
Schoolmaster
The classroom
Slot machines
Cash lost in ‘em
Statue of liberty
Built to stay free
Snooze alarms
Awa’s No more Z’s
The Morse code
Here come dots
Vacation times
I’m not as active
Western Union
No wire unsent
George Bush
He bugs Gore
Parishioners
I hire parsons
The earthquakes
That queen shake
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?)

Assertion: true-false statement (comment) asserting a belief
about (the current state of) your program.
// x is the sum of 1..n <- asserts a specific relationship
between x and n
x ? n 1

x ? n 0

Assertions help prevent bugs by helping you keep track of what
you’re doing …
and they help track down bugs by making it easier to check
belief/code mismatches.

Assertions can help with bugs in loops: initialization errors,
termination errors, and processing errors.

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

// x = sum of 1..n-1
n= n + 1;
// x =  sum of 1..n
precondition
1 2 3 4 5 6 7 8
x contains the sum of these (6)
postcondition
1 2 3 4 5 6 7 8
x contains the sum of these (10)

What statement do you put here
so that segment is correct? (if
precondition is true, execution
of segment should make
postcondition true.)
A. x= x + 1;
B. x= x + n;
C. x= x + n+1;
D. None of A, B, C
E. I can’t figure it out

Solving a problem

// x = sum of 1..n
n= 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.)
A. x= x + 1;
B. x= x + n;
C. x= x + n+1;
D. None of A, B, C
E. I can’t figure it out

Invariants: another type of assertion
An invariant is an assertion about the variables that is true before and
after each iteration (execution of the repetend).
for (int i= 2; i <= 4; i= i + 1) {
    x= x + i*i;
}
// invariant
Invariant:
x = sum of squares of 2..i-1
A. i= 2;
B. x= x + i*i;
C. x= x + i*i;
D. i= i + 1;
E. I can’t figure it out
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. Write loop invariant.
5. Figure out any initialization.
6. Implement the repetend (Process k).

// Process b..c
Initialize variables (if necessary) to make invariant true.
// Invariant: range b..k-1 has been processed
for (int k = b; k <= c; k = k+1) {
    // Process k
}
// Postcondition: range b..c has been processed

Finding an invariant

// Store in double variable v the sum
// 1/1 + 1/2 + 1/3 + 1/4 + 1/5 + ... + 1/n
v = 0;
// invariant: v = sum of 1/i for i in 1..k-1
for (int k = 1; k <= n; k = k +1) {
    Process k;
}
// v = 1/1 + 1/2 + ... + 1/n
What is the invariant? 1 2 3 ... k-1 k k+1 ... n

Finding an invariant

// set x to no. of adjacent equal pairs in s[0..s.length()-1]
// invariant:
for (int k = 0; k < s.length(); k = k +1) {
    Process k;
}
// x = no. of adjacent equal pairs in s[0..s.length()-1]
k: next integer to process. What is the invariant?
Which ones have been processed?
A. 0..k
B. 0..k-1
C. a..k
D. a..k-1
E. None of the above

Being careful

// { String s has at least 1 char }
// Set c to largest char in String s

// inv: c is largest char in s[0..k-1]
for (int k = ; k < s.length(); k = k +1) {
    // Process k;
}
// c = largest char in s[0..s.length()-1]
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.

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