CS100J 17 October, 2006
The while loop and assertions

Start reading chapter 7 on loops.
The lectures on the ProgramLive CD can be a big help.

Some anagrams

A decimal point	I'm a dot in place	Animosity	Is no amity
Debit card	Bad credit	Desperation	A rope ends it
Dormitory	Dirty room	Funeral	Real fun
Schoolmaster	The classroom	Slot machines	Cash lost in 'em
Statue of liberty	Built to stay free	Snooze alarms	Alas! 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!)

The while loop: syntax

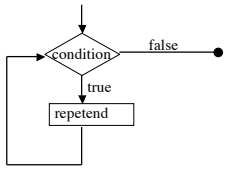
while (*<condition>*)
  <repetend>

<condition>: a boolean expression.

<repetend>: a statement.

while (*<condition>* {
  sequence of declarations and statements
}

BUT: We always make the *<repetend>* a block.



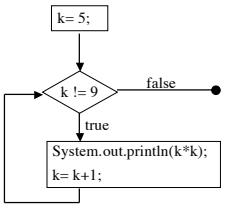
The while loop

```
System.out.println(5*5);
System.out.println(6*6);
System.out.println(7*7);
System.out.println(8*8);

int k= 5;
while ( k != 9) {
    System.out.println(k*k);
    k= k+1;
}
```

To execute the while loop:
(1) Evaluate condition k != 9; if false, stop execution.
(2) Execute the repetend.
(3) Repeat again from step (1).

Trace execution of the loop: Study section 7.1.2 shows you how to "trace" execution of a loop.



The while loop: syntax

while (*<condition>*)
  <repetend>

<condition>: a boolean expression.

<repetend>: a statement.

while (*<condition>* {
  sequence of declarations and statements
}

BUT: We always make the *<repetend>* a block.

for-loop

```
int k;
for (k= 5;  k != 9; k= k+1) {
    System.out.println(k*k);
}
```

Equivalent while-loop

```
int k= 5;
while ( k != 9) {
    System.out.println(k*k);
    k= k+1;
}
```


The while loop: syntax

```
// Set c to the largest character in String s.
// Precondition: s has at least one character
int n= s.length() - 1;
c= s.charAt(0);
//  c is the largest character in s[0..k-1]
for (int k= 1; k < n; k= k+1) {
   if (s.charAt(k) > c)
         c= s.charAt(k);

}
// c is the largest character in s[0..n-1]
```

```
int n= s.length() - 1;
int k= 1; c= s.charAt(0);

while (k < n) {
    if (s.charAt(k) > c)
          c= s.charAt(k);
    k= k+1;
}
```


Understanding assertions

What value of x or n makes this assertion true?

x is the sum of 1..n — x [ ] n [1]

h..k is the set of values
h, h+1, h+1, …, k.
Example: 3..6 is 3, 4, 5, 6.
In this notation, we require h ≤ k–1.
h..h–1 is the empty set.

x [ ] n [3]
x [ ] n [4]
x [ ] n [0]
x [0] n [ ]

What's the simplest solution to this? Make the range 1..n as small as possible.

x [ ] n [ ]

Understanding assertions

What value of x makes this assertion true?

b = "some value in 5..9 divides x"

b **false** x 1

b x 16

b x 10

Below, fill in the assignments so that the assertion is true afterward.

k= ;

b= ;

// { **b = "some value in 2..k divides x"** }

b x ?

k

h..k is the set of values h, h+1, h+1, …, k.
In this notation, we require h ≤ k–1. h..h–1 is the empty set.

Understanding assertions

Fill in the space after the assignment to k with an if-statement so that the assertion remains true after execution.

// { **b = "some value in 2..k divides x"** }

k= k + 1 ;

Make up an example, if it helps, but learn to work in this more abstract setting

b ? x ?

// { **b = "some value in 2..k divides x"** }

k ?

h..k is the set of values h, h+1, h+1, …, k.
In this notation, we require h ≤ k–1. h..h–1 is the empty set.

Understanding assertions

// { nothing has been printed }

k= ;

// {squares of 5..k-1 printed}

Fill in the assignments on this page so that the assertions following them are true.

k= ;

// {All chars in String s[0..k–1] are '$'}

// { k ≥ 1 }

k= ;

c= ;

// // {c is the smallest character in s[0..k–1]}

Hint: make the range 0..k–1 as small as possible

Understanding assertions

Suppose this assertion is true:
x = sum of 1..k
Under what extra condition is this one true?
x = sum of 1..n

Put your answer here

Suppose this assertion is true:
x = sum of h..10
Under what extra condition is this one true?
x = sum of 1..10

Put your answer here

Suppose this assertion is true:
no value in h..k divides x
Under what extra condition is this one true?
no value in 2..n–1 divides x

Put your answer here

Understanding assertions

	0	1	2	3	4	5	6	7	8
v	X	Y	Z	X	A	C	Z	Z	Z

This is a list of Characters

	0		3		k		8
v	≥ C		?		all Z's		

k 6

This is an assertion about v and k. It is **true** because chars of v[0..3] are greater than 'C' and chars of v[6..8] are 'Z's.

	0		3		k		8
v	≥ C		?		all Z's		

k 5

	0			k		8
v	≥ C			all Z's		

k 6

	0			k		8
v	≥ W		A	C	all Z's	

k 4

Indicate whether each of these 3 assertions is true or false.

Suppose we have this while loop, with initialization:

initialization;
while (B) {
  repetend
}

We add the postcondition and also show where the invariant must be true:

initialization;
// invariant: P
while (B) {
  // { P }
  repetend
  // { P }
}
// { P }
// { Result R }

The four loopy questions

Second box helps us develop four loopy questions for developing or understanding a loop:

1. How does loop start? Initialization must truthify inv P.

2. When does loop stop?

At end, P and B are true, and these must imply R. Find !B that satisfies P && R => R.

3. Make progress toward termination? Put something in repetend to ensure this.

4. How to keep invariant true? Put something in repetend to ensure this.