CS 125 - Lecture 9

Objectives:

- String and character programs; operator precedence;
- To do : Finish MP1; Read course notes; Turing's Craft exercises, start MP2 tomorrow.

1. Java Operator Precedence Table:

Precedence	Operator	Type	Associativity
15	() [] ·	Parentheses Array subscript Member selection	Left to Right
14	++ --	Unary post-increment Unary post-decrement	Right to left
13	++ -- + - ! ~ (*type*)	Unary pre-increment Unary pre-decrement Unary plus Unary minus Unary logical negation Unary bitwise complement Unary type cast	Right to left
12	* / %	Multiplication Division Modulus	Left to right
11	+ -	Addition Subtraction	Left to right
10	<< >> >>>	Bitwise left shift Bitwise right shift with sign extension Bitwise right shift with zero extension	Left to right
9	< <= > >= instanceof	Relational less than Relational less than or equal Relational greater than Relational greater than or equal Type comparison (objects only)	Left to right
8	== !=	Relational is equal to Relational is not equal to	Left to right
7	&	Bitwise AND	Left to right
6	^	Bitwise exclusive OR	Left to right
5	|	Bitwise inclusive OR	Left to right
4	&&	Logical AND	Left to right
3	||	Logical OR	Left to right
2	? :	Ternary conditional	Right to left
1	= += -= *= /=	Assignment Addition assignment Subtraction assignment Multiplication assignment Division assignment	Right to left

Memorize this? … Better yet, use parentheses!

2. Operator precedence practice:

3 + 2 + 5*4

Evaluates to 3 + 2 + 20

(multiplication before addition)

3 + 2 + 20

Evaluates to…

5 + 20

25

(most operators work left to right)

Evaluate the following statement using Java's precedence rules

boolean r;

r= ! true || false != false;

boolean r;

r=5 + 1 % 3 < 2 && 3 < 2 == false;

3. Test your Java knowledge:

What does concatenation mean? How is it implemented in Java?

T/F Thoroughly commenting your code will significantly speed your program's execution

Are the following valid Java statements? If so, what will each print?

i) `TextIO.putln("Result :"+2+3);`

ii) `TextIO.putln( 2+3+"Result" );`

iii) `intvalue=((2+(3/10)+5.0)<10) == true;`

Give three elements of good style mentioned in the reading:

4. Useful String methods 'subroutines' from pre-lecture reading ch2.3

s1. ____________ (s2) returns true if s1 and s2 have the same character sequence.

s1. ____________ () the number of characters in s1.

s1. ______________ (N) returns a *char* at position N

s1. ________________ (N,M) returns a string from N^th (inclusive) position up to but excluding M^th position.

s1.____________ (s2) returns an integer. If s2 occurs as a substring of s1, then the returned value is the starting position of that substring. Otherwise, the returned value is -1.

s1. _____________________ () returns a new string with lower case letters converted to upper case.

CS 125 - Lecture 9

6. Write the following programs (don't waste time writing the opening Class and Program statements, or writing out entire prompt text below):

```
Please enter a string where the first and last letters are
the same:
You typed “abbA”
You win!
```

7. **Analyze this: How many dots are printed?**

```
public static void main(String[] args) {
   int a = 0;
   int b;
   while (a < 20) {
      a += 2;
      b = 1;
      while (b < 16) {
         TextIO.put('.');
         b = b * 2;
      }
      TextIO.putln(a);
   }
}
```

5. Write the following programs (don't waste time writing the opening Class and Program statements, or writing out entire prompt text below):

```
Enter a string with exactly 5 characters. You typed:1234
Try again!
Enter a string with exactly 5 characters. You typed:12345
Yes!
```

How to think about variables...

Fixed value: The role of a variable or an attribute is a fixed value, if its value is not changed after initialization.

Stepper: Stepper goes through a succession of values in some systematic way, predictable succession of values.

Most-recent holder: The value of a most-recent holder is the latest gone through value of a certain group or simply the latest input value.

Most-wanted holder: The value of a most-wanted holder is the "best" or otherwise the most-wanted value out of the values gone through so far. THE most-wanted can mean, for example, the smallest or the biggest number or a number closest to a certain value.

Gatherer: The value of a gatherer accumulates all the values gone through so far.

Follower: A follower always gets the old value of another known variable or attribute as its new value.

One-way flag: A one-way flag has two possible values but cannot get its original value anymore after it has been once changed.

Temporary: The value of a temporary is always needed only for a very short period.

Objectives:

- String and character programs; operator precedence;
- To do : Finish MP1; Read course notes; Turing’s Craft exercises, start MP2

1. Fill in the missing the code and fix any errors you notice.
Update the code so that it keeps asking for a password until a good password is entered.

```
_____________ done = false;
__________________________
TextIO. ________________________ ( “Prompt the user: New
password? 10 or more characters, mixed case, no spaces" );

___________________ = TextIO. _____________

____________ short = __________________; // true if too short
____________ noUpperCase= _____________
____________ hasSpaces = _____________
____________ badPass = short || noUpperCase && hasSpaces;

if( _____________) }
     TextIO.putln("Bad password - try again.");
}
TextIO.putln("Password accepted, thanks.");
```

2. **Strings Activity**

Comparison:

- Why do we use `s1.equals(s2)` What's 'wrong' with `s1 == s2` ?

- How many String objects are created here?

```
String s1 = "Bonjour".toUpperCase() + "monde";
String sAnother = s1;
TextIO.put(sAnother);
```

- Fix the program below to determine the average word length in a text file ‘speech.txt’. For simplicity, you can assume one word per line and no punctuation.

```
public class Speech {
     public static void main(String[] args)
     {

          TextIO.readFile(“speech.txt");


          while( ! TextIO.eof() ) {

          }
```

3. **[Piazza *Challenge*]** Complete the following and post to Piazza:

```
TextIO.putln("Acme crossword solver. What do you know so far?");
String s = TextIO.getln(); // User enters “g??g??"
```

// Now print a series of words from dictionary file (https://courses.engr.illinois.edu/CS125/lecture-notes/dictionary.txt) that match the user's pattern ('?' means any character). The file has one word per line. Inform the user how many words matched.

2. **Strings Activity**

- Why do we use `s1.equals(s2)` What's 'wrong' with `s1 == s2` ?

- How many String objects are created here?

```
String s1 = "Bonjour".toUpperCase() + "monde";
String sAnother = s1;
TextIO.put(sAnother);
```