# CS180 Spring 2011 Exam 1 *with solutions*, 28 February, 2011

*Prof. Chris Clifton*

**Turn Off Your Cell Phone.** Use of any electronic device during the test is prohibited.

Time will be tight. If you spend more than the recommended time on any question, **go on to the next one**. If you can't answer it in the recommended time, you are either going in to too much detail or the question is material you don't know well. You can skip one or two parts and still demonstrate what I believe to be an A-level understanding of the material.

For multiple choice questions, please circle the letter next to your choice. If a question says "circle all that are true", then you will get one point for each correct one you circle, and lose one point for each incorrect one you circle (you will not receive less than 0 points for a question.)

*I would expect someone with an A-level knowledge of the material to score at least 33, someone with a B level knowledge to score at least 24. If you received less than 14, you should probably come talk to me.*

## 1 Types (1 minute, 2 points)

Which of the following is **not** a legal statement (or sequence of statements). Circle only one.

A
```
boolean b;
b = true;
```

B
```
String s;
s = true;
```

C `int i = 3;`

D `String s = "3";`

E
```
double f;
f = 3;
```

**Only B is not legal - the boolean value true cannot be assigned to a String.**

## 2 Methods (7 minutes, 6 points)

Given the following method:

```
public int longerThan ( String s, int len )
{
  return s.length() - len;
}
```

*Note: The exam as given had "string", not "String", for the first argument type. If you showed evidence that you noticed this, then several other answers are possible.*

### 2.1 Calling a Method (2 minutes, 2 points)

Assume you are calling it from another method in the same class. One of the following statements *might* not compile; all the others do.

Circle the one that might be incorrect.

A
```
if ( longerThan ("abc", 3) > 0 )
    System.out.println("The String abc is longer than 3");
```

B `System.out.println(longerThan(abc, 3));`

C `int a = 2 * longerThan("abc", 2);`

D `longerThan("a", 4);`

**B is correct. If you showed evidence that you noticed the argument wasn't a String, then any answer consistent with your answer to 2.2 is acceptable.**

### 2.2 Calling a Method part 2 (2 minutes, 2 points)

Explain your answer to 2.1. For full credit, describe conditions under which it would compile.

**B will only compile if abc is declared as a variable of type String (String abc = "";)**

*If you noted that longerThan takes a string, not a String, there are many other acceptable answers.*

Scoring: 1 point for showing evidence you knew why it might not compile, 1 for conditions under which it would.

### 2.3 Calling a Method part 3 (3 minutes, 2 points)

Assume that you had a `main` method in the same class as the `longerThan` method. I claim that none of the statements in question 2.1 could be used in the `main` method. Explain why.

**You cannot call a non-static method without having an object.**

**Main is static, so it is not running in an object. Hence a call to longerThan would not have an object.**

*Again, if you noted that longerThan takes a string, not a String, there are many other acceptable answers.*

## 3 Looping (10 minutes, 7 points)

Given the following for loop:

```
sum = 0;
for ( int i = 0; i < 10; sum++ )
  i = i + 2;
```

### 3.1 Code Execution (3 minutes, 2 points)

What is the value of sum after executing the above code (circle only one)?

A 0

B 4

**C 5**

D 9

E 10

### 3.2 Proper Use of Loops (2 minutes, 2 points)

The above for loop is a poor style for using a for loop. Explain why (briefly).

**The variable modified in the third part of the for should be the variable that is checked for termination** (1 point), **the variable i should not be modified elsewhere (otherwise a while loop would be more appropriate)** (1 point), **variables other than that used to check termination of the loop should not be modified in the for statement** (1 point), and **the variable modified in the for statement should not be used after the loop** (1 point).

### 3.3 Better Style (3 minutes, 3 points)

Rewrite the above loop (you may use a for, while, or do-while) in a better style. `sum` should have the same value when the loop is complete as the above code.

```
sum = 0;
int i = 0;
while ( i < 10 ) {
  i = i + 2;
  sum++;
}
```

or

```
sum = 0;
for ( int i = 0; i < 10; i = i + 2)
  sum++;
```

Scoring: One point for having the same outcome using a loop to increment sum; one point each for correcting style errors mentioned part 3.2.

## 4 Data Abstraction (2 minutes)

Which of the following statements are true regarding interfaces and data abstraction (circle all that are true)? You will receive one point for each correct answer circled, and lose one point for each incorrect answer circled.

A An interface can define the body of the functions within it.

**B An interface can be used to provide an abstraction of some functionality.**

**C Defining a class which describes the attributes and behavior of a student is an example of data abstraction.**

D Data abstraction requires an interface to be designed and implemented.

## 5 Classes and Interfaces (5 minutes)

Consider the following interface and class signatures:

```
public interface I
{
  public void f();
  public void g();
}

public interface J
{
  public void p();
  public void q();
}
```

```
public class A implements J
{
  //
}

public class B implements I
{
  //
}
```

### 5.1 Short Answer: Implementing an Interface (2 minutes, 2 points)

Minimally, how many methods must be written inside class A? Name the methods.

**2 methods** (1 point), **p and q** (1 point).

## 5.2 Multiple Choice: Creating Objects with Classes and Interfaces (2 minutes)

Which of the following declaration, object instantiation, and assignments are valid. (Circle all that are legal; you get one point for each correct, and lose one point for each wrong one.)

A `I i = new I();`

B `I i = new B();`

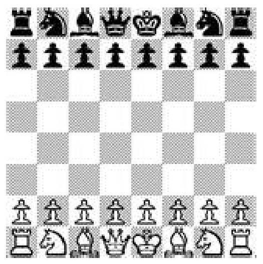
C `J j = new A();`

D `A a = new J();`

E `B b = new B();`

**B, C, and E are correct.**

## 6 Chess Program (25 minutes, 16 points)



For the next few questions, you will work with the following program that simulates a chess board. You don't have to know how to play chess to do this (and if you do, forget what you know, as our chess will have simplified rules.)
A chessboard has rows and columns. The columns are named with letters. (e.g., 'a' or 'b'). The rows are numbered starting with 1.

```
public interface ChessPiece
{

  boolean move(char toColumn, int toRow);
    /* If toColumn and toRow is a legal place
     * for this piece to move to, move it there
     * and return true.  Otherwise don't move,
     * and return false. */

  void showPosition();
    // Print the current position on System.out
}

public class Pawn implements ChessPiece
{
  private char atColumn;
  private int  atRow;

  public Pawn(char startingColumn)
  {
    atRow = 2; // Pawns always start in row 2.
    atColumn = startingColumn;
  }

  public void showPosition()
  {
    System.out.println("Pawn is at " + atColumn + atRow);
  }
}
```

```
public class PlayChess
{
  public static void main( String args[] )
  {
    ChessPiece qp = new Pawn('d');
    if ( qp.move('d', 3) )
      System.out.println("You made a legal move");
    else
      System.out.println("Bad move");
    // Done
  }
}
```

## 6.1 Multiple Choice: Compilation (2 minutes, 2 points)

When you try to compile the above code, it gives the following error:

A variable qp might not have been initialized

B incompatible types. found: int, required: boolean

**C Pawn is not abstract and does not override abstract method**

D move(char,int) in ChessPiece cannot be applied to (int,int)

E cannot find symbol : constructor Pawn(char,int)

*All are plausible errors, but only C occurs in the given code. For example, the constructor "Pawn(char,int)" doesn't exist, but since nobody tries to call such a constructor, it doesn't matter.*

A: 1; B, C: 0

## 6.2 Multiple Choice: Objects (2 minutes, 2 points)

Assume the error in Question 6.1 has been fixed, and the program compiles and runs without errors. When the program reaches the line "// Done" in the main method, what objects are there in the system?

A No objects have been created.

B 1: a ChessPiece object.

**C 1: a Pawn object.**

D 2: a ChessPiece object and a Pawn object.

E 3: a ChessPiece object, a PlayChess object, and a Pawn object.

*ChessPiece is an interface, you can't create an object for an interface.*

## 6.3 Short Answer: Using Interfaces (3 minutes, 3 points)

Assume that I have written a Knight class that implements the ChessPiece interface. I have also created a knight object kk. Write a statement that will move the object kk to column f row 3. *Hint: You do not write any code that goes in the Knight class – that is already written, and you just have to use it.* For full credit, you should also check if this was a legal move and print a message if it was not.

**if (! kk.move('f',3) ) System.out.println("Not a legal move.");**

Scoring: 1 for using the kk object, 1 for correct arguments, 1 for checking result.

## 6.4 Code: Implementing Interfaces (10 minutes, 9 points)

The Pawn class does not have the move method that is required by the ChessPiece interface. Write that method. Assume that a pawn only moves one space forward at a time; that is, it always stays in the same column, and moves to the next row (e.g., if a pawn is at column *d* row *2*, it can only move to column *d* row *3*; an attempt to move it anywhere else is not allowed.) Everything else you need to know is contained in the code above.

```
public boolean move(char toColumn, int toRow)
{
  if ( toColumn == atColumn && toRow == atRow + 1 ) {
    atRow = toRow;
    return true;
  }
  else
    return false;
}
```

Scoring: 1 for a "move" method, 1 for public, 1 for correct return type, 1 for correct arguments, 1 for check on column, 1 for check on row, 1 for returning boolean, 1 for updating row, 1 for logically correct.

*A common error was to use "toColumn.equals(atColumn)". Since a char is a primitive type, and not an object, you cannot call a method on it.*