1. **[8]** Create declarations for each of the following. *You do not need to provide any constructors or method definitions*.

   (a) The instance variables of a class to hold information on a "Minesweeper" cell: whether it has a bomb in it, whether it is has been clicked, and how many bomb neighbours it has. The class header has been provided for you.

   ```
   public class MineSweeperCell {









           . . .
   }
   ```

   (b) The instance constant and class variable required to add a vehicle number (VIN) to the Car class. The Cars will be numbered consecutively (that is, first Car is 1, second Car is 2, and so on).

   ```
   public class Car {









           . . .
   }
   ```

2. **[20]** Suppose we start declaring a Rectangle class as follows:

```
public class Rectangle {
    private double width;
    private double height;
}
```

a) Write a **constructor** for the class, which takes as its two arguments the initial width and height of the Rectangle. If a negative width or height is provided, the constructor replaces it with a zero.

b) Write a **setter** for the Rectangle's width. If a negative width is requested, the method does nothing (no error message).

c) Write a method that returns the **area** of the Rectangle. (The area of a rectangle is its height times its width.)

d) Write a toString method for this class. The String is of the form “*w*x*h* Rectangle”, where w is the width and h is the height. (For example, “5x20 Rectangle”).

e) Write a method to *draw* the Rectangle using * characters. For example, a 2x5 Rectangle would appear as:

```
*****
*****
```

3. 10. **[6]** Suppose we have created classes for Sections, Professors, and Rooms. Suppose further than each Section has a lecturer (who is a Professor), each Professor has an office (which is a Room), and each Room has a building (which is a String). We have also created a Section object in the variable mySection. Assuming that getters have been written for each of these instance variables, which of the following commands/expressions will compile without an error message?

a)	mySection.getLecturer()	OK	Error
b)	mySection.getOffice()	OK	Error
c)	mySection.getOffice().toUpperCase()	OK	Error
d)	mySection.getLecturer().getOffice()	OK	Error
e)	mySection.getLecturer().getOffice().getBuilding()	OK	Error
f)	mySection.getLecturer().getOffice().toUpperCase()	OK	Error

4. **[4]** For each of the following declaration/partial declarations, indicate whether it is a class variable (CV), class constant (CC), instance variable (IV), method (M), or constructor (C). *There is one of each shown.*

   a) _____ private static double bad_name = 3.0;

   b) _____ private String bad_name = “Hello”;

   c) _____ public bad_name() {...}

   d) _____ public static final int bad_name = 5;

   e) _____ public void bad_name() {...}

5. **[6]** Write a method (printArray) to print all the elements of an array of integers on a single line. There must be spaces between the numbers, but no punctuation (commas, brackets, *etc*.). (For example, the array below is printed as `67 34 100`.)

67	34	100

6. **[8]** Write a method that creates an array of integer values from 1 to n, where n is given to the method. For example, the call makeArray(5) returns the array:

1	2	3	4	5

7. **[12]** Multiple Choice: select the *best available* answer from the options shown.

- If MyData is a class, and list1 is a variable of type MyData, then list1 is called
  a) a class variable.
  b) a field.
  c) an instance variable.
  d) an object.
  e) a record.

- The return type of the constructor for class MyData (containing a String and two int values) would be
  a) int
  b) MyData
  c) String
  d) void
  e) (constructors have no return type)

- When a variable is declared static, that means that
  a) anyone can access that variable.
  b) each object of that class has its own copy of that variable.
  c) it is shared by all instances of this class.
  d) its value will never change.
  e) no other class (including the class with the main method) can access it.

- Suppose we have already created `drawBox(int w, int h, char edge, int indent)`, which draws a `w` by `h` box indented `indent` characters on the screen, using `edge` as the character at the edge of the box. We want to write *another* drawBox method, `drawBox(int w, int h)` that draws a box on the screen, indented zero characters, using a star ('*') as the character at the edge of the box. The body of that method consists entirely of:
  a) drawBox(int w, int h, char '*', int 0);
  b) drawBox(int w, int h, char edge, int indent);
  c) drawBox(w, h);
  d) drawBox(w, h, '*', 0);
  e) drawBox(w, h, "*", 0);

- Suppose we want to add an equals method to the class Whosit, so we can create if statements starting like `if (whosit1.equals(whosit2))`. The header (or interface) for the method would be:
  a) public boolean equals(Whosit one, Whosit other)
  b) public boolean equals(Whosit other)
  c) public equals(Whosit one, Whosit other)
  d) public equals(Whosit other)
  e) public static boolean equals(Whosit other)

- The class `Dohickey` has two constructors: one with no parameters, and one with a single, int parameter. The declaration `new Dohickey`() creates an object equivalent to the one created by `new Dohickey(100)`. The body of the parameterless constructor would be:
  a) Dohickey = 100;
  b) Dohickey(100);
  c) this();
     Dohickey = 100;
  d) this(100);
  e) this.Dohickey = 100;

- When there are two methods with the same name in the same class, differing only in their parameters, the method name is said to be
  a) overloaded.
  b) overrated.
  c) overruled.
  d) overwritten.
  e) (the situation described is not legal in Java)

- example, the `i` in `a[i]`) is called
  a) a component
  b) an element
  c) an index
  d) (a or b)
  e) (b or c)

- What is the output of the following code?

```
int[] a = new int[10];
System.out.println(a[10]);
```

  a) 0
  b) 10
  c) (nothing will be printed because the code will not compile)
  d) (nothing will be printed, because the program will crash)
  e) (we don't know what'll be printed, because the elements were not initialized)

- Instance variables should be declared _____ unless they are _____.
  a) final; public.
  b) final; static.
  c) private; final.
  d) private; static.
  e) public; final.
  f) static; public.

- The maximum possible grade for every Student is the same, so the variable holding that value should be declared
  a) final.
  b) int.
  c) int[].
  d) private.
  e) public.
  f) static.

- The command to make c a copy of the array a is:
  a) Arrays.copy(a, c);
  b) Arrays.copyOf(a) = c;
  c) c = Arrays.copy(a);
  d) c = Arrays.copyOf(a);
  e) c = Arrays.copyOf(a, a.length);

- Which of the following types **cannot** be made into an array type by adding []?
  a) int
  b) Scanner
  c) String
  d) String[]
  e) (any of the above can be made into the base type of an array)