Signature _____ **Name _____**

cs11f ____ **Student ID _____**

CSE 11
Final
Fall 2009

Page 1 _____ (24 points)

Page 2 _____ (7 points)

Page 3 _____ (13 points)

Page 4 _____ (34 points)

Page 5 _____ (13 points)

Page 6 _____ (15 points)

Page 7 _____ (23 points)

Page 8 _____ (18 points)

Page 9 _____ (19 points)

Total _____ (166 points = 158 base points + 8 points EC [5%])
(100%)

(Partial) Operator Precedence Table

Operators				Associativity
!	++	-- (pre & post inc/dec)		right to left
*	/	%		left to right
+	-			left to right
<	<=	>	>=	left to right
==	!=			left to right
&&				left to right
||				left to right
=				right to left

1) Which of the following <u>are not</u> valid Java identifiers? (Circle your answer(s).)

```
1stJavaClass      My_First_Java_Class     Java1             CSE11Is#1

CSE11             CSE-11                  My1stJavaClass    double
```

2) Using the operator precedence table above, evaluate each expression and state what gets printed. Remember short-circuit evaluation with && and ||.

```
int i = 1, j = 2, k = 3, m = 2;

System.out.println( !( k >= m ) );                          _________

System.out.println( j <= i || j == m && k <= m );           _________

System.out.println( i >= 1 && !(j != 4) );                  _________

System.out.println( !(i > 4 && j <= 6) == i >= 4 || j > 6 ); _________
```

3) What gets printed?

```
int a = 3, b = 6;

System.out.println(  -1  +  ++a  *  5  +  17  %  5  );     __________

System.out.println(  6  +  b++  -  5  /  9  +  4  );       __________
```

4) What gets printed?

```
public class Question4
{
   public static void main( String[] args )
   {
      final int MAX = 9;
      int i = 4, j = 8;
      for (i = 6; i <= MAX; ++i )
      {
         j = i;
         while ( j < MAX )
         {
            --j;
            System.out.println( i + " " + j );
            j += 2;
         }
      }
      System.out.println( i + " " + j );
   }
}
```

5) What gets printed as a result of the call `F5( -1, 3 )`? _____

```
public void F5( int a, int b )
{
  if ( (a >= 0) || (b <= a) )
  {
    if ( a <= b )
    {
      System.out.println( "A" );
    }
    else
    {
      System.out.println( "B" );
    }
  }
  else if ( (a < 0) && (b < 0) )
  {
    System.out.println( "C" );
  }
  else
  {
    System.out.println( "D" );
  }
}
```

Using only the values -2 and -1, give an example of values passed as arguments to F5() that would result in the method printing "B". The values -2 and -1 can be in any order and may be repeated (you do not need to use both values – both arguments may be the same value).

```
F5( _____ , _____ );
```

6) Assume the following code is defined:

```
public class NumberHolder
{
  private int number;

  public NumberHolder()
  {
    this.number = 0;
  }

  public void setNum(int n)
  {
    this.number = n;
  }

  public int getNum()
  {
    return this.number;
  }
}
```

What is the output of the following code using the above class definition?

```
NumberHolder a = new NumberHolder();
a.setNum( 5 );
NumberHolder b = a;
System.out.println( a.getNum() + " " + b.getNum() );  _____
b.setNum( 2 );
System.out.println( a.getNum() + " " + b.getNum() );  _____
```

7) Given the following class definitions:

```
public class Person
{
  public Person() { ... }
  public void print() { System.out.println( "Person" ); }
  public void printAll( Person[] list )
  {
    for ( int i = 0; i < list.length; ++i )
      list[i].print();
  }
}

public class Student extends Person
{
  public void print() { System.out.println( "Student" ); }
}
```

Assume the method printAll() is called with an array of length 5, and than none of the five elements of the array is null. Which of the following statements best describes what will happen, and why? Circle correct answer.

A. The word Person will be printed five times since the type of the array parameter is Person.
B. The word Person will be printed five times since printAll is a method of the Person class.
C. The word Student will be printed five times since the print method was overridden by the Student class.
D. For each of the five objects in the array, either the word Person or the word Student will be printed, depending on the type of the objects in the array list.
E. If the array actually contains objects of type Person, then the word Person will be printed five times; otherwise, a runtime error will occur.

8) Complete the following method which is intended to return the index of the last occurrence of `value` in the array `numbers` or -1 if `value` is not in the array `numbers`.

```
public static int findLastOccurrence( int[] numbers, int value )
{
  for ( int i = _____________________________________; _____________________________; --i )
  {
    if ( ____________________________________________________ )
    {
      return _____________ ;
    }
  }
  return _____________;
}
```

9) What does the following method print as a result of the call `F9( 10 )`?

```
public void F9( int x )
{
  for ( int y = 0; y <= x; y = y + 2 )
  {
    System.out.print( y + " " );
  }

  System.out.println();

  if ( x > 0 )
  {
    F9( x – 2 );
  }

  System.out.print( x + " " );
}
```

10) Indicate whether each of the following parts of a Java program is (A-H) and where in the Java Runtime Environment each part lives (1-3)

A) Class (static) variable
B) Instance variable
C) Static method
D) Instance method
E) Local variable
F) Formal Parameter
G) Constructor
H) Class definition

1) The Class Area
2) The Heap
3) Stack Frame in the Runtime Stack

	Java program part (Answer A-H in this column)		Java Runtime area (Answer 1-3 in this column)
`public class F10`	_____	F10	
`{`			
`  private char actor;`	_____	actor	_____
`  public F10() { }`	_____	F10	_____
`  public void setActor( char ch ) { actor = ch; }`	_____	ch	_____
	_____	setActor	_____
`  public static int cling;`	_____	cling	_____
`}`			
`public class SomeOtherClass`	_____	SomeOtherClass	
`{`			
`  private int cling;`	_____	cling	_____
`  public static void main( String[] args )`	_____	args	_____
`  {`			
	_____	main	_____
`    char toon = '?';`	_____	toon	_____
`    F10 ref1;`	_____	ref1	_____
`    ref1 = new F10();`		(where ref1 is pointing)	_____
`    SomeOtherClass ref2 = new SomeOtherClass();`	_____	ref2	_____
`    // Other Code ... possibly changes the value in toon`		(where ref2 is pointing)	_____
`        ...    // *** Location 1 ***`			
`  }`			
`  public char fubar( char tester ) { ... }`	_____	tester	_____
	_____	fubar	_____
`}`			

Write a single statement that could appear above at the line marked `//*** Location 1 ***` that passes the value of **toon** to **fubar** and puts the return value of **fubar** into the variable **actor** in the object referenced by ref1.

Write a single statement that could appear above at the line marked `//*** Location 1 ***` that puts the value of **cling** in class F10 into the variable **cling** in the SomeOtherClass object referenced by **ref2**.

11) Given the following partial class definition fill in the body of the constructors using the supplied comments as a guide.

```
public class Foo2 extends Foo1
{
  private Fubar var2;
  private double var3;

  public Foo2( int var1, Fubar var2, double var3 )
  {

    _____________________________________        // Explicitly invoke super class (Foo1) constructor
                                                 //   passing the parameter var1.

    _____________________________________        // Initialize the double instance variable to the
                                                 //   parameter var3.

    _____________________________________        // Initialize the Fubar instance variable by invoking
                                                 //   the copy ctor for Fubar with parameter var2.
  }                                              // Assume a copy ctor for Fubar is defined.

  public Foo2()
  {
                                                 // Call same class ctor passing in 42 for var1,
    _____________________________________        //   a new Fubar object invoking its no-arg ctor for
                                                 //   var2, and 80.86 for var3.
  }                                              // Assume a no-arg ctor for Fubar is defined.
}
```

Assuming class Foo1 has only one constructor, and based on the comments and your code above, write the full constructor that must be in class Foo1.

```
public class Foo1
{
  private __________ var1;









}
```

12) Consider the following code segment:

```
int[] a = { 1, 2, 3 };
int[] b = { 1, 2, 3 };
int[] c = a;
```

After this code executes, which of the following expressions would evaluate to true? Circle correct answer in the box to the right.

I. a.equals(b)
II. a == b
III. a == c

A. I only
B. II only
C. III only
D. I and II only
E. I and III only
F. II and III only
G. I, II, and III

13) Given the following definitions:

```
public abstract class MyPet
{
  public abstract String speak();
}
```

```
public class Puppy extends MyPet
{
  private static final String
                 PUPPY_SPEAK = "Bark";

  public Puppy()
  {
    // ctor initialization here
  }

  public String speak()
  {
    return PUPPY_SPEAK;
  }

  public void sleep( int time )
  {
    // puppy sleeps for time seconds
  }
}
```

```
public class Kitty extends MyPet
{
  private static final String
                 KITTY_SPEAK = "Meow";

  public Kitty()
  {
    // ctor initialization here
  }

  public String speak()
  {
    return KITTY_SPEAK;
  }

  public void wag()
  {
    // kitty wags its tail
  }
}
```

And the following variable definitions:

```
Puppy puppy;
Kitty kitty;
MyPet pet;
```

Indicate which are valid Java statements. Consider each statement executed sequentially in the order it appears.

A) Invalid Java statement – Compiler Error
B) Valid Java statement – No Compiler Error

Hint: What does the compiler know about any reference variable at compile time (vs. run time)?

```
kitty = new Kitty();          _______

puppy = new Puppy();          _______

pet = kitty;                  _______

pet.speak();                  _______

pet.wag();                    _______

pet.sleep( 3000 );            _______

kitty = pet;                  _______

pet = new MyPet();            _______

pet = puppy;                  _______

pet.speak();                  _______

( (Puppy) pet).wag();         _______

( (Puppy) pet).sleep( 3000 ); _______

puppy = pet;                  _______

puppy = kitty;                _______

kitty.wag();                  _______
```

14) What is the output produced by the following program? (Hint: draw stack frames)

```
public class Mystery
{
  public static void main( String[] args )
  {
    Mystery ref = new Mystery();

    System.out.println( ref.mystery( 9 ) );
  }

  public int mystery( int a )
  {
    int b = a + 3;
    int c = a - 3;

    if ( c > 0 )
    {
      System.out.println( a + " " + b + " " + c );
      c = b + mystery( a - 2 );
      System.out.println( a + " " + b + " " + c );
    }
    else
    {
      c = a + b;
      System.out.println( "Stop!" );
      System.out.println( a + " " + b + " " + c );
    }

    return c;
  }
}
```

Output

Given the following class definitions:

```
abstract class Animal
{
  private String name;
  public Animal() { this( "Animal" ); }
  public Animal( String name ) { this.name = name; }
  public String toString() { return name; }
  public abstract String speak();
}

class Cat extends Animal
{
  public Cat() {}
  public Cat( String name ) { super( name ); }
  public String speak() { return "Meow"; }
}

class Tiger extends Cat
{
  public Tiger() { this( "Tigger" ); };
  public Tiger( String name ) { super( "Tiger " + name ); }
  public String speak( String name ) { return name + speak(); }
}

class BigTiger extends Tiger
{
  public BigTiger() { super( "Big" ); }
  public BigTiger( String name ) { super( name ); }
  public String speak( String name ) { return "Sorry " + name; }
}

class Lion extends Cat
{
  public String speak() { return "Lion " + louder(); }
  public String louder() { return "Louder Lion " + super.speak(); }
}

public class F15
{
  public static void main( String[] args )
  {
    Animal a;

    a = new Tiger();
    System.out.println( a + " says " + ( (Tiger) a).speak( "Elin " ) );

    a = new BigTiger( "Woods" );
    System.out.println( a + " says " + ( (Tiger) a).speak( "fans") );

    a = new Lion();
    System.out.println( a + " says " + a.speak() );

    a = new Cat( "Tiger" );
    System.out.println( a + " says " + a.speak() );
  }
}
```

15) What gets printed when this program is run?

Given the following class definitions for class Foo, class Fubar, and class FubarTest:

```
public class Foo
{
  public Foo( int x, int y )
  {
    this();
    System.out.println( "Foo ctor #1" );
  }

  public Foo()
  {
    System.out.println( "Foo ctor #2" );
  }

  public String toString()
  {
    System.out.println( "Foo.toString" );
    return "Foo.toString";
  }
}
```

```
public class Fubar1 extends Foo
{
  public Fubar1( int x, int y, int z )
  {
    super( x, y );
    System.out.println( "Fubar ctor #1" );
  }

  public Fubar1( int x, int y )
  {
    this( x, y, -99 );
    System.out.println( "Fubar ctor #2" );
  }

  public String toString()
  {
    System.out.println( "Fubar.toString" );
    return super.toString() + " + " +
           "Fubar.toString";
  }
}
```

```
public class FubarTest
{
  public static void main( String[] args )
  {
    Foo ref = new Fubar1( 5, 10 );

    System.out.println( "-----" );

    System.out.println( ref.toString() );
  }
}
```

17) What is the output when we run FubarTest as in
java FubarTest

18) What is Rick's favorite beer? _____

Java supports single inheritance of _____ using the keyword _____.

Composition provides a(n) _____ relationship while inheritance provides a(n) _____ relationship.

When assigning a variable of type double to a variable of type int, Java requires you to use a _____ on the double variable.

A(n) _____ can contain only public abstract methods and public static final constants.

Java supports multiple inheritance of _____ using the keyword _____.

To check for exact type equivalence, call _____ on the two objects and check if the resulting references are the same with == .

Scratch Paper

Scratch Paper