EXAMINATION INSTRUCTIONS

NAME: _____

PERSON NUMBER: _____

SIGNATURE: _____

This examination has 6 pages. Check that you have a complete paper.
Each candidate should be prepared to produce, upon request, his or her SUNY/UB card.
This examination has 5 questions. Answer all questions.
You have 60 minutes to complete this examination. Use your time accordingly.

READ AND OBSERVE THE FOLLOWING RULES:

- Names are pre-printed on the exam booklets. Ensure that you have YOUR exam.
- Sign, using your usual signature, in the space provided on the back cover.
- All of your writing must be handed in. This booklet must not be torn or mutilated in any way, and must not be taken from the examination room.
- Show all of your work in arriving at an answer, unless instructed otherwise. Partial credit will be awarded as appropriate.
- Candidates are not permitted to ask questions of the invigilators, except in cases of supposed errors or ambiguities in examination questions.
- CAUTION – Candidates guilty of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action.
  - Making use of any books, papers or memoranda, calculators or computers, audio or visual cassette players, or other memory aid devices, other than those explicitly authorised by the examiners.
  - Speaking or communicating with other candidates.
  - Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

---------- DO NOT WRITE BELOW THIS LINE! ----------

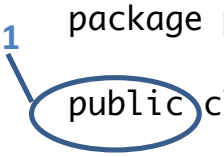
Q1	Q2	Q3	Q4	Q5	TOTAL	%
/10	/10	/10	/10	/10	/50	**/100**

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors.

Circle, and identify by number, one ***and only one*** example of each of the following items in the code below. If you believe no example exists, write "*no example*" next to that item in the list. To show you how I want the question answered, *the first one is done for you.*

1. access control modifier
2. expression
3. class body
4. fully qualified name
5. argument
6. assignment statement




```
package part1;

public class Farm {

	private example1.BarnYard _yard;

	public Farm() {
		_yard = new example1.BarnYard();
	}

	public void addOneMovingChicken() {
		example1.Chicken c = new example1.Chicken();
		_yard.addChicken(c);
		c.start();
	}

	public void addOneMovingPig() {
		example1.Pig p = new example1.Pig();
		_yard.addPig(p);
		p.start();
	}

	public void addOneMovingButterfly() {
		example1.Butterfly b = new example1.Butterfly();
		_yard.addButterfly(b);
		b.start();
	}

}
```

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors.

Circle, and identify by number, one ***and only one*** example of each of the following items in the code below. If you believe no example exists, write "*no example*" next to that item in the list. To show you how I want the question answered, *the first one is done for you.*

1. access control modifier
2. expression
3. class body
4. fully qualified name
5. argument
6. assignment statement

1

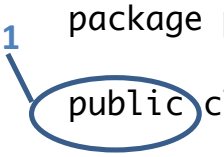
```
package part1;

public class Farm {

    private example1.BarnYard _yard;

    public Farm() {
        _yard = new example1.BarnYard();
    }

    public void addOneMovingChicken() {
        example1.Chicken c = new example1.Chicken();
        _yard.addChicken(c);
        c.start();
    }

    public void addOneMovingPig() {
        example1.Pig p = new example1.Pig();
        _yard.addPig(p);
        p.start();
    }

    public void addOneMovingButterfly() {
        example1.Butterfly b = new example1.Butterfly();
        _yard.addButterfly(b);
        b.start();
    }

}
```

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors.

Circle, and identify by number, one ***and only one*** example of each of the following items in the code below. If you believe no example exists, write "*no example*" next to that item in the list. To show you how I want the question answered, *the first one is done for you.*

1. access control modifier
2. expression
3. class body
4. fully qualified name
5. argument
6. assignment statement




```
package part1;

public class Farm {

	private example1.BarnYard _yard;

	public Farm() {
		_yard = new example1.BarnYard();
	}

	public void addOneMovingChicken() {
		example1.Chicken c = new example1.Chicken();
		_yard.addChicken(c);
		c.start();
	}

	public void addOneMovingPig() {
		example1.Pig p = new example1.Pig();
		_yard.addPig(p);
		p.start();
	}

	public void addOneMovingButterfly() {
		example1.Butterfly b = new example1.Butterfly();
		_yard.addButterfly(b);
		b.start();
	}

}
```

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors.

Circle, and identify by number, one ***and only one*** example of each of the following items in the code below. If you believe no example exists, write "*no example*" next to that item in the list. To show you how I want the question answered, *the first one is done for you.*

1. access control modifier
2. expression
3. class body
4. fully qualified name
5. argument
6. assignment statement

```
package part1;

public class Farm {

	private example1.BarnYard _yard;

	public Farm() {
		_yard = new example1.BarnYard();
	}

	public void addOneMovingChicken() {
		example1.Chicken c = new example1.Chicken();
		_yard.addChicken(c);
		c.start();
	}

	public void addOneMovingPig() {
		example1.Pig p = new example1.Pig();
		_yard.addPig(p);
		p.start();
	}

	public void addOneMovingButterfly() {
		example1.Butterfly b = new example1.Butterfly();
		_yard.addButterfly(b);
		b.start();
	}

}
```

1

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors.

Circle, and identify by number, one ***and only one*** example of each of the following items in the code below. If you believe no example exists, write "*no example*" next to that item in the list. To show you how I want the question answered, *the first one is done for you.*

1. access control modifier
2. expression
3. class body
4. fully qualified name
5. argument
6. assignment statement

```
package part1;

public class Farm {

	private example1.BarnYard _yard;

	public Farm() {
		_yard = new example1.BarnYard();
	}

	public void addOneMovingChicken() {
		example1.Chicken c = new example1.Chicken();
		_yard.addChicken(c);
		c.start();
	}

	public void addOneMovingPig() {
		example1.Pig p = new example1.Pig(); or p=new example1.Pig();
		_yard.addPig(p);
		p.start();
	}

	public void addOneMovingButterfly() {
		example1.Butterfly b = new example1.Butterfly(); (ditto)
		_yard.addButterfly(b);
		b.start();
	}

}
```

1

Question 2 [10 points, 2 points each]

For each of the following questions, select the **BEST** answer from the available choices.

[2 POINTS] Where in memory are method invocation records stored?
a) secondary storage
b) heap
c) static region
d) runtime stack

[2 POINTS] Where in memory are static variables stored?
a) secondary storage
b) heap
c) static region
d) runtime stack

[2 POINTS] From which area of memory does 'new' allocate space?
a) secondary storage
b) heap
c) static region
d) runtime stack

[2 POINTS] Which of the following has the same value as 1110_2?
a) 7_{10}
b) 14_{10}
c) 28_{10}
d) 56_{10}

[2 POINTS] What is the value of a 'new' expression, like new part2.Farm() ?
a) a reference
b) an instance
c) a variable
d) an object

Question 3 [10 points, 2 points each]

The code sample given below is correct: it compiles without errors.

```
package exam1;
public class FamilyFarm {
      public FamilyFarm() {
            example1.BarnYard enclosure;
            enclosure = new example1.BarnYard();
            example1.Chicken daisy;
            daisy = new example1.Chicken();
            example1.Pig wilbur;
            wilbur = new example1.Pig();
            enclosure.addChicken(new example1.Chicken());
            enclosure.addPig(wilbur);
      }
}
```

ANSWER THE QUESTIONS BELOW:

a) How many variables are declared in the code shown above? [2 points]

3

For questions **(b) through (e)** **assume that the class exam1.FamilyFarm is instantiated once.**

b) How many example1.BarnYard objects are created? [2 points]

1

c) How many example1.Chicken objects are created? [2 points]

2

d) How many exam1.FamilyFarm objects are created? [2 points]

1

e) How many objects are added to BarnYard enclosure? [2 points]

2

Question 4 [10 points – 10 points perfect, 7 points essentially correct but with small mistakes, 3 points clearly wrong but with some correct elements, 0 points for anything else]

Study the following partial class definition (the omitted parts are not relevant to the question). Complete the definition of a parameterless void method named swap so that it interchanges the values of the the two instance variables _a and _b.

```
package exam;

public class Question4 {

	private String _a;
	private String _b;

	public Question4() {
	 // body of constructor not relevant to question
	}

	// define your swap method below:



	public void swap() {
		String temp;
		temp = _a;
		_a = _b;
		_b = temp;
	}




}
```

Question 5 [10 points]

Complete each of the following sentences. Choose **the best answer** for each sentence from the phrases given below. You may use each phrase at most once; not all phrases will be used.

- A variable declaration determines a/the ____(n) type_____ and a name for a variable.
- When carrying out an assignment statement a/the ____(i) expression_____ on the right of the a/the __(b) assignment_____ operator is evaluated first, and the resulting value is stored in a/the ___(k) variable_____ on the left.
- When performing a method call, the _(a) arguments_____ are evaluated and the resulting values are assigned to the corresponding __(d) parameters_____.
- The 'new' operator allocates memory on the ___(l) heap_____ to hold the representation of an object.
- ____(j) local_____ variables are stored on the runtime stack.
- An __(c) object_____ is an instance of a __(e) class_____.

PHRASES:

~~a) arguments~~
~~b) assignment~~
~~c) object~~
~~d) parameters~~
~~e) class~~
f) method
g) instance
h) stack
~~i) expression~~
~~j) local~~
~~k) variable~~
~~l) heap~~
m) private
~~n) type~~
o) name
p) value
q) reserved word
r) package