SPRING 2013 - CENG 2002 MIDTERM

1) (40 points, each correct match 4 points) For the following PL concepts at right, provide an example from the code given at left. Mark the relevant part(s) of the code and then use the question letter to denote the related concept. 3 of the concepts don't have a corresponding example. If you are sure which ones, mark these also.

```
class Point {
  public int x, y;
}
class Rectangle {
  public float w, h;
  public Point topLeft;
}
class Question1 {
 public void f(int z) {
   float pi = 3.14;
   Rectangle rect = new Rectangle(...);
 }
 public int g(int z) {
   if (z<=0) return 1;
   else return z*g(z-1);
 }
 public static void main(String[] args) {
  boolean married = true;
  int[] arr = { 2, 4, 6, 8, 10 };
  int a=0,b=0,c=0,i,j;
  float w = 1234567.89;
  i = married ? arr.length : 3;
  while(i--) {
    System.out.println(arr[i]);
    int c = arr[i] * 2;
    f(g(c));
    f(c--);
  }
  arr = { 1, 3, 5, 7, 9 };
  Rectangle r = new Rectangle(...);
  Point p1 = new Point(...);
  r.topLeft = p1;
 }
}
```

a. A method call
b. A heap variable
c. A recursive declaration
d. A reference parameter
e. A conditional expression
f. A binding which results in hiding
g. A construction used during binding
h. A variable whose type is an abstract type
i. A formal parameter whose type is composite
j. A total update(j1) and a partial update(j2)
k. A simple variable whose lifetime is not program's run-time
l. A literal expression whose type has cardinality 2
m. An expression with a side effect which is also an actual parameter

2) (40 points, each correct answer 5 points) Suppose that we have two programming languages called A and B which are Object Oriented and have a Java-like syntax. These languages have the following semantic differences:

- All arithmetic operators in A are left associative but in B they are right associative.
- In A the assignment operator employs reference semantics for composite values but in B copy semantics is used. Both languages use copy semantics for primitive values.
- Functions are first class values in B but not in A.
- In A, function parameters are pass by value but in B they are pass by reference.
- In A, dynamic dispatch is allowed but in B it is not allowed.
- A only allows static arrays but in B dynamic arrays are possible.
- Language A allows mutually recursive definitions but in B they are not allowed.
- In A there is a deallocator but in B there is none.

For each of the following code snippets if there are printLn statements give the output for both languages. Otherwise, specify in which language the code will be valid or invalid.

```
// Example 2A
void  f(... a, ... b) {
...
x = a(b);
...
}
```

```
// Example 2B
int s = 10;
...
while(...) { ... s++ ...}
int[] arr = new int[s];
```

```
// Example 2C
class Car { ...
  public int tires { return 4; }
... };
class Bus extends Car { ...
  public int tires { return 8; }
... };
...
Car myCar; a = myCar.tires());
Bus myBus; b = myBus.tires());
myCar = myBus;
c = myCar.tires();
```

```
// Example 2D
int f(int a, int b) {
  b *= 2; return --a * b;
}
...
int x=3, y=10;
printLn(f(x,y));
printLn(x + “ “ +  y);
```

```
// Example 2E
class Point {
  public int x,y;
  public Point(int a, int b) {
    x=a; y=b; }
}
...
int a=3, b=7;
a=b;
b=9;
Point p1 = new Point(a,b);
Point p2 = p1;
p1.x = 123;
printLn(p1.x + “,” + p1.y);
printLn(p2.x + “,” + p2.y);
```

```
// Example 2F
class Point { ... }
...
Point p = new Point(...);
....
del(p); // destroys the heap variable
```

```
// Example 2G
printLn(37 % 10 % 4);
```

```
// Example 2H
class Course {
  public string code;
  public int credits;
  Instructor teacher;
}
class Instructor {
  public string name;
  public Course[] courses;
}
```

3) (10 points) Rewrite the following code segment by replacing the conditional command with conditional expressions.

```
if (x>0) {
  print(“Yes”);
  return 10;
}
else {
  print(“No”);
  return 20;
}
```

4) (10 points) Suppose that we defined the following new types in an Ada-like language:

```
type Color is (red, green, blue);
type Coord is range 1..8;
type Board is array (Coord, Coord) of Color;
type Row is record
      r: Coord;
      colorOdd: Color;
      colorEven: Color;
end record;
function checkRow(r: Row) return Boolean is ..
```

a) Give the cardinality of user defined type Board.
b) Give the cardinality of function checkRow.