AP Computer Science Practice Midterm Answers Name:\_\_\_\_\_\_\_\_

Multiple Choice

1. Consider the following code segment.

for (int k = 0; k < 20; k = k + 2)

{

if (k % 3 == 1)

System.out.print(k + " ");

}

What is printed as a result of executing the code segment?

(a) 4 16

**(b) 4 10 16**

(c) 0 6 12 18

(d) 1 4 7 10 13 16 19

(e) 0 2 4 6 8 10 12 14 16 18

2. Consider the following code segment.

ArrayList<String> list = new ArrayList<String>();

list.add("P");

list.add("Q");

list.add("R");

list.set(2, "s");

list.add(2, "T");

list.add("u");

System.out.println(list);

What is printed as a result of executing the code segment?

(a) [P, Q, R, s, T]

(b) [P, Q, s, T, u]

**(c) [P, Q, T, s, u]**

(d) [P, T, Q, s, u]

(e) [P, T, s, R, u]

3. Consider the following instance variable and method.

private ArrayList<Integer> nums;

/ \*\* **Precondition**: nums.size . 0

\*/

public void numQuest()

{

int k = 0;

Integer zero = new Integer(0);

while (k < nums.size())

{

if (nums.get(k).equals(zero))

nums.remove(k);

k++;

}

}

Assume that List nums initially contains the following

Integer values.

[0, 0, 4, 2, 5, 0, 3, 0]

What will List nums contain as a result of executing numQuest ?

(a) [0, 0, 4, 2, 5, 0, 3, 0]

(b) [4, 2, 5, 3]

(c) [0, 0, 0, 0, 4, 2, 5, 3]

(d) [3, 5, 2, 4, 0, 0, 0, 0]

**(e) [0, 4, 2, 5, 3]**

6. A car dealership needs a program to store information about the cars for sale.

For each car, they want to keep track of the following information: number of doors (2 or 4), whether the car has air conditioning, and its average number of miles per gallon. Which of the following is the best design?

**(a) Use one class, Car, which has three data fields:**

**int numDoors, boolean hasAir, and**

**double milesPerGallon.**

(b) Use four unrelated classes: Car, Doors, AirConditioning, and MilesPerGallon.

(c) Use a class Car which has three subclasses: Doors, AirConditioning, and MilesPerGallon.

(d) Use a class Car, which has a subclass Doors, with a subclass AirConditioning, with a subclass MilesPerGallon.

(e) Use three classes: Doors, AirConditioning, and MilesPerGallon, each with a subclass Car.

14. Consider the following method.

/\*\* **Precondition**: x ≥ 0

\*/

public void mystery(int x)

{

System.out.print(x % 10);

if ((x / 10) != 0)

{

mystery(x / 10);

}

System.out.print(x % 10);

}

Which of the following is printed as a result of the call mystery(1234) ?

(a) 1441

(b) 3443

(c) 12344321

**(d) 43211234**

(e) Many digits are printed due to infinite recursion.

15. Consider the following two classes.

public class Dog

{

public void act()

{

System.out.print("run");

eat();

}

public void eat()

{

System.out.print("eat");

}

}

public class UnderDog extends Dog

{

public void act()

{

super.act();

System.out.print("sleep");

}

public void eat()

{

super.eat();

System.out.print("bark");

}

}

Assume that the following declaration appears in a client program.

Dog fido = new UnderDog();

What is printed as a result of the call fido.act() ?

(a) run eat

(b) run eat sleep

(c) run eat sleep bark

**(d) run eat bark sleep**

(e) Nothing is printed due to infinite recursion.

19. Consider the following code segment.

int[][] mat = new int[3][4];

for (int row = 0; row < mat.length; row++)

{

for (int col = 0; col < mat[0].length; col++)

{

if (row < col)

mat[row][col] = 1;

else if (row == col)

mat[row][col] = 2;

else

mat[row][col] = 3;

}

}

What are the contents of mat after the code segment has been executed?

(a) { {2 1 1}, {3 2 1}, {3 3 2}, {3 3 3}}

(b) { {2 3 3}, {1 2 3}, {1 1 2}, {1 1 1}}

(c) { {2 3 3 3}, {1 2 3 3}, {1 1 2 3}}

**(d) { {2 1 1 1}, {3 2 1 1}, {3 3 2 1}}**

(e) { {1 1 1 1}, {2 2 2 2}, {3 3 3 3}}

20. Consider the following methods.

public List<Integer> process1(int n)

{

ArrayList<Integer> someList = new ArrayList<Integer>();

for (int k = 0; k < n; k++)

someList.add(new Integer(k));

return someList;

}

public ArrrayList<Integer> process2(int n)

{

ArrayList<Integer> someList = new ArrayList<Integer>();

for (int k = 0; k < n; k++)

someList.add(k, new Integer(k));

return someList;

}

Which of the following best describes the behavior of process1 and process2?

**(a) Both methods produce the same result and take the same amount of time.**

(b) Both methods produce the same result, and process1 is faster than process2.

(c) The two methods produce different results and take the same amount of time.

(d) The two methods produce different results, and process1 is faster than process2.

(e) The two methods produce different results, and process2 is faster than process1.

22. Consider the following declaration for a class that will be used to represent

points in the *xy*-coordinate plane.

public class Point

{

private int myX; // coordinates

private int myY;

public Point()

{

myX = 0; myY = 0;

}

public Point(int a, int b)

{

myX = a; myY = b;

}

// ... other methods not shown

}

The following incomplete class declaration is intended to extend the above class so that points can be named.

public class NamedPoint extends Point

{

private String myName;

// constructors go here ... other methods not shown

}

**Consider the following proposed constructors for this class.**

I. public NamedPoint()

{

myName = "";

}

II. public NamedPoint(int d1, int d2, String name)

{

myX = d1; myY = d2; myName = name;

}

III. public NamedPoint(int d1, int d2, String name)

{

super(d1, d2); myName = name;

}

Which of these constructors would be legal for the NamedPoint class?

(a) I only

(b) II only

(c) III only

**(d) I and III**

(e) II and III

Free Response Question



**Part (a)**

public class Cat extends Pet 1

{

 public Cat(String name)

 {

 super(name); 2

 }

 public String speak()

 {

 return "meow";

 }

}

**Notes:**

1. Just follow the given template for the Dog class.
2. This constructor must call the Pet constructor and pass name to it.

**Part (b)**

public class LoudDog extends Dog 1

{

 public LoudDog(String name)

 {

 super(name);

 }

 public String speak()

 {

 String s = super.speak(); 2

 return s + s; 3

 }

}

**Notes:**

1. Do not define new fields in this class.  The myName field and getName method are already defined in Pet.
2. Obtain the "speak" string by calling Dog's speak.
3. Or simply
4. return super.speak() + super.speak();

**Part (c)**

 // postcondition: for each pet in the kennel, its name followed

 // by the result of a call to its speak method

 // has been printed, one line per Pet

 public void allSpeak()

 {

 for (int i = 0; i < petList.size(); i++)

 {

 Pet pet = petList.get(i); 1

 System.out.println(pet.getName() + " " + pet.speak());

 }

 }