A+ Computer Science

AP REVIEW

2012 FR QUESTIONS

Provided by A+ Computer Science

Visit us at

www.apluscompsci.com

Full Curriculum Solutions

M/C Review Question Banks

Live Programming Problems

Tons of great content!

www.facebook.com/APlusComputerScience

Free Response

-Read all 4 questions before writing anything

-answer the easiest question 1st

-most times question 1 is the easiest

-see if part B calls part A and so on

-many times part C consists of A and B calls

-write something on every question

-write legibly / use PENCIL!!!!!!!!!!

-keep track of your time

Free Response

-When writing methods

  -use parameter types and names as provided

  -do not redefine the parameters listed

  -do not redefine the methods provided

  -return from all return methods

  -return correct data type from return methods

Free Response

-When writing a class or methods for a class

  -know which methods you have

  -know which instance variables you have

  -check for public/private on methods/variables

  -return from all return methods

  -return correct data type from return methods

Free Response

-When extending a class

  -know which methods the parent contains

  -have the original class where you can see it

  -make sure you have super calls

  -check for public/private on methods/variables

  -make super calls in sub class methods as needed

Free Response

-When extending abstract / implementing interface

  -know which methods the parent contains

  -have the original class where you can see it

  -make sure you have super calls

  -check for public/private on methods/variables

  -make super calls in sub class methods as needed

  -implement all abstract methods in sub class

Free Response Topics

ArrayList of References / Objects
– get,set,remove,add,size – levels of abstraction

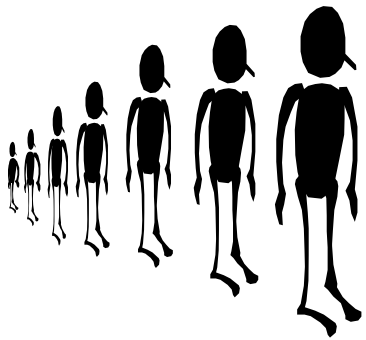
Matrix / 2 D Array
– nested loops, GridWorld (grid)

GridWorld or Make a Class
– location, actor, bug, critter, grid, super, abstract

String / Array Question
– find biggest, find smallest, etc.

ArrayList

A typical ArrayList question involves putting something into an ArrayList and removing something from an ArrayList.



ArrayList

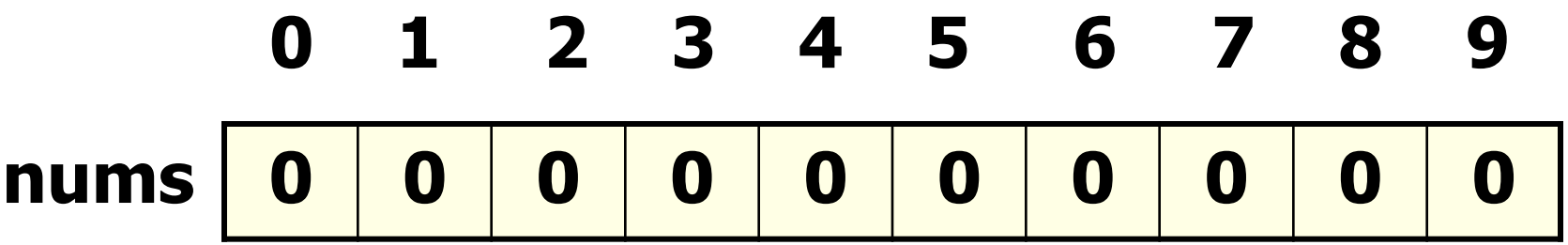
Arraylist is a class that houses an array.

An ArrayList can store any type.

All ArrayLists store the first reference at spot / index position 0.

ArrayList

```
int[] nums = new int[10];        //Java int array
```



An array is a group of items all of the same type which are accessed through a single identifier.

ArrayList

frequently used methods

Name	Use
add(item)	adds item to the end of the list
add(spot,item)	adds item at spot – shifts items up->
set(spot,item)	put item at spot z[spot]=item
get(spot)	returns the item at spot return z[spot]
size()	returns the # of items in the list
remove()	removes an item from the list
clear()	removes all items from the list

```
import  java.util.ArrayList;
```

ArrayList

```
List<String> ray;
ray = new ArrayList<String>();
ray.add("hello");
ray.add("whoot");
ray.add("contests");
out.println(ray.get(0).charAt(0));
out.println(ray.get(2).charAt(0));
```

OUTPUT

```
h
c
```

ray stores String references.

ArrayList

```
int spot=list.size()-1;
while(spot>=0)
{
   if(list.get(spot).equals("killIt"))
      list.remove(spot);

   spot--;

}
```

ArrayList

```
for(int spot=list.size()-1; i>=0; i--)
{
   if(list.get(spot).equals("killIt"))
      list.remove(spot);
}
```

ArrayList

```
int spot=0;
while(spot<list.size())
{
   if(list.get(spot).equals("killIt"))
      list.remove(spot);
   else
      spot++;
}
```

2012 Question 1 - part A

```
public void addClimb( String peakName, int climbTime )
{
  ClimbInfo c = new ClimbInfo( peakName, climbTime );
  climbList.add( c );
}
```

You must know ArrayList!

2012 Question 1 - part B

```
public void addClimb( String peakName, int climbTime )
{
  ClimbInfo c = new ClimbInfo( peakName, climbTime );

  for( int i = 0; i < climbList.size(); i++ )  {
    int x = peakName.compareTo( climbList.get(i).getName() );
    if( x <= 0 ) {
      climbList.add( i , c );
      break;
    }
  }

  if( sz == climbList.size() )
    climbList.add( c );
}
```

You must know

ArrayList!

2012 Question 1 - part C

NO
YES

You must know ArrayList!

Abstract / Interfaces

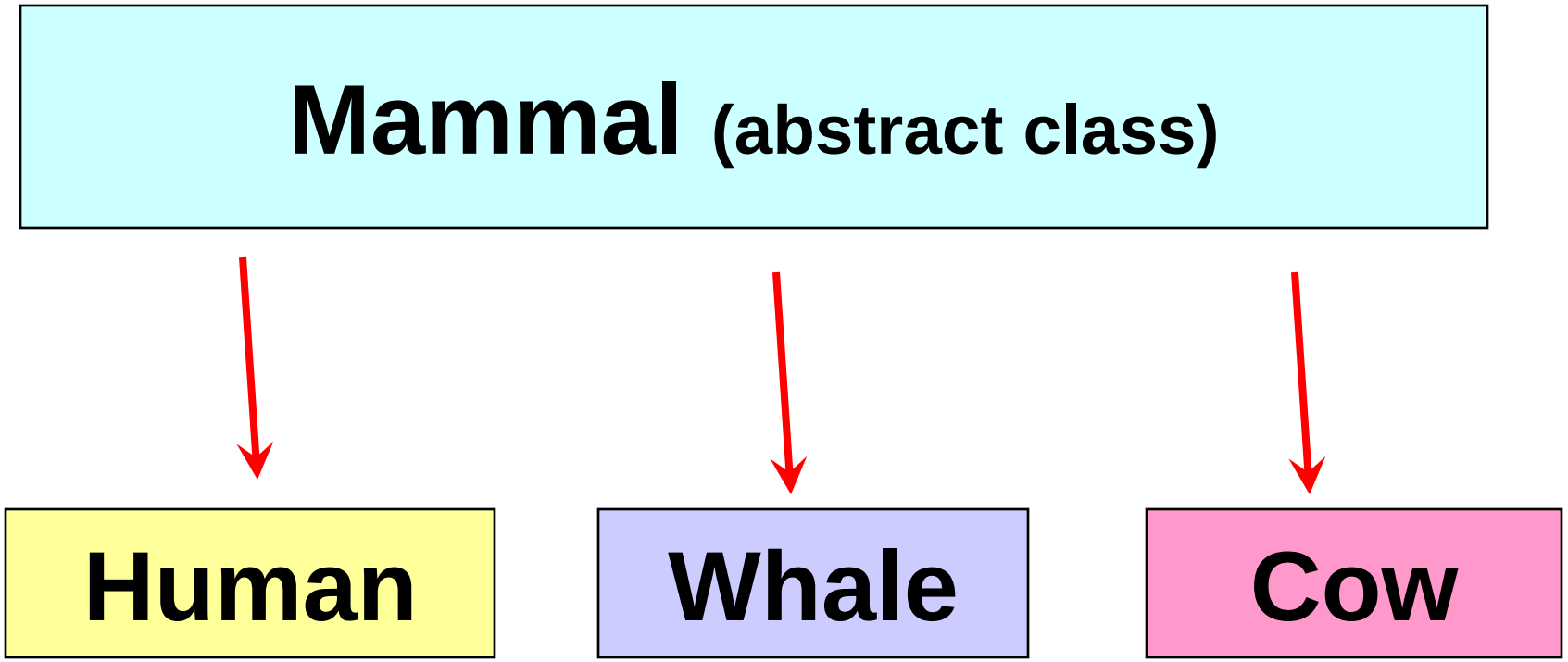
A typical Abstract/Interface question requires that a class be written that extends the abstract class or implements the interface and that all abstract method(s) be implemented.

Abstract / Interfaces

Abstract classes are used to define a class that will be used only to build new classes.

No objects will ever be instantiated from an abstract class.

Abstract / Interfaces



Abstract / Interfaces

Any sub class that extends a super abstract class must implement all methods defined as abstract in the super class.

Abstract / Interfaces

```
public abstract class APlus
{
   public APlus(int x)
      //constructor code not shown

   public abstract double goForIt();

   //other fields/methods not shown
}
```

Pet
Item

Abstract / Interfaces

```
public class PassAPTest extends APlus
{
  public PassAPTest(int x)
  {
    super(x);
  }

  public double goForIt()
  {
    double run=0.0;
    //write some code   -   run = x*y/z
    return run;
  }

  //other fields/methods not shown
}
```

```
public abstract class APlus
{
  public APlus(int x)
    //constructor code not shown

  public abstract double goForIt();

  //other fields/methods not shown
}
```

Abstract / Interfaces

```
public interface Exampleable
{
   int writeIt(Object o);
   int x = 123;
}
```

Methods are public abstract!
Variables are public static final!

Abstract / Interfaces

```
public interface Exampleable
{
   public abstract int writeIt(Object o);
   public static final int x = 123;
}
```

Methods are public abstract!
Variables are public static final!

Abstract / Interfaces

An interface is a list of abstract methods that must be implemented.

An interface may not contain any implemented methods.

Interfaces cannot have constructors!!!

Abstract / Interfaces

Interfaces are typically used when you know what you want an Object to do, but do not know how it will be done.

If only the behavior is known, use an interface.

Abstract / Interfaces

Abstract classes are typically used when you know what you want an Object to do and have a bit of an idea how it will be done.

If the behavior is known and some properties are known, use an abstract class.

Hodge Podge

One question on the A test free response is usually a random question that is hard to predict.

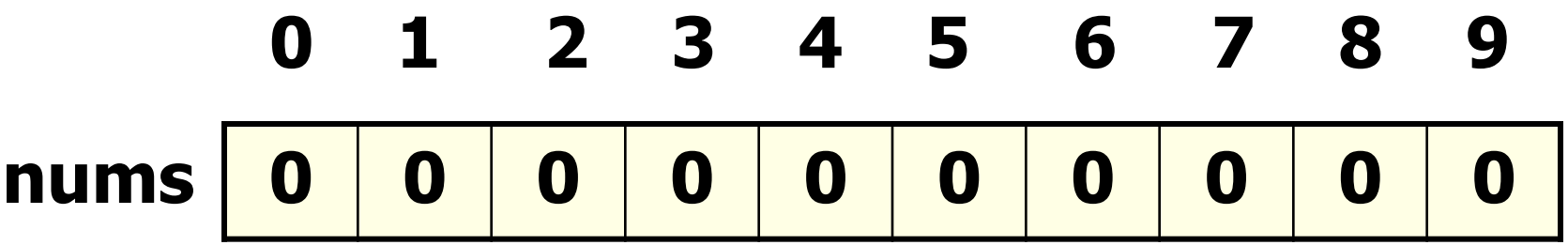
CustomerSort
Robot
Reservation

Hodge Podge

This question usually involves an array and many times has sorting and searching components.

Hodge Podge

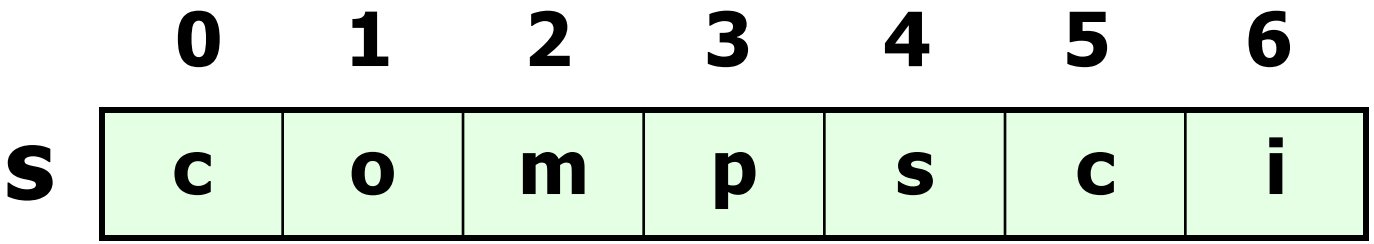
```
int[] nums = new int[10];          //Java int array
```



An array is a group of items all of the same type which are accessed through a single identifier.

Hodge Podge

String s = "compsci";



A string is a group of characters.
The first character in the group is at spot 0.

String

frequently used methods

Name	Use
substring(x,y)	returns a section of the string from x to y not including y
substring(x)	returns a section of the string from x to length-1
length()	returns the # of chars
charAt(x)	returns the char at spot x
indexOf(c)	returns the loc of char c in the string, searching from spot 0 to spot length-1
lastIndexOf(c)	returns the loc of char c in the string, searching from spot length-1 to spot 0

String

frequently used methods

Name	Use
equals(s)	checks if this string has same chars as s
compareTo(s)	compares this string and s for >,<, and ==
trim()	removes leading and trailing whitespace
replaceAll(x,y)	returns a new String with all x changed to y
toUpperCase()	returns a new String with uppercase chars
toLowerCase()	returns a new String with lowercase chars

Hodge Podge

```
String sent = "alligators rule";
String find = "gato";

System.out.println( sent.indexOf( find ) );
System.out.println( sent.indexOf( "dog" ) );
System.out.println( sent.substring( 3 , 6 ) );
System.out.println( sent.substring( 6 ) );
```

OUTPUT

```
4
-1
iga
tors rule
```

2012 Question 3 - part A

```
public int findHorseSpace( String name )
{
 for( int i = 0; i < spaces.length; i++ )
 {
    if( spaces[i] != null )
      if(name.equals(spaces[i].getName())
        return i;
  }
 }
 return -1;
}
```

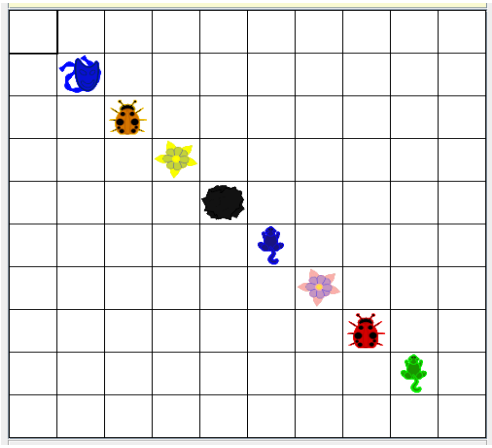
2012 Question 3 - part B

```
public void consolidate( )
{
 Horse[] tmp = new Horse[ spaces.length ];
 int loc = 0;
 for( Horse h : spaces)
   if( h != null )
  {
     tmp[ loc ] = h;
     loc++;
   }

 spaces = tmp;
}
```

Matrices

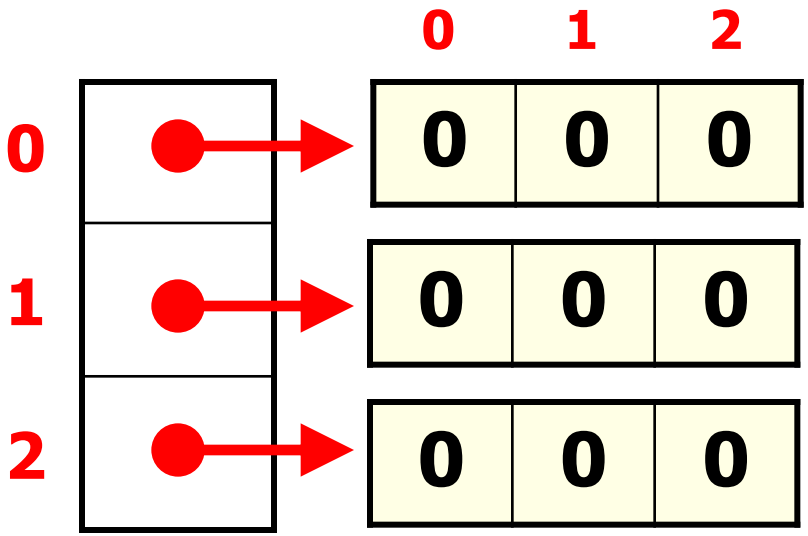
One question on the A test free response will require you to manipulate a 2-dimensional array or a GridWorld grid.



Matrices

A matrix is an array of arrays.

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



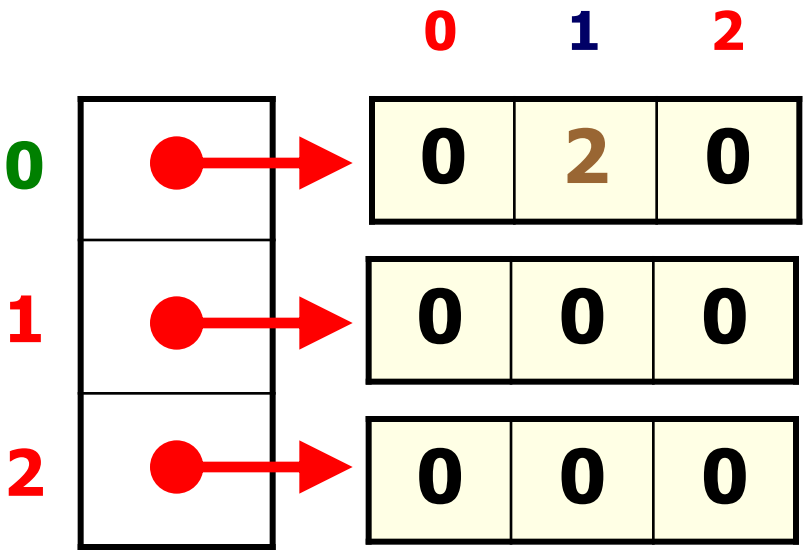
Matrices

A matrix is an array of arrays.

```
int[][] mat = new int[3][3];
mat[0][1]=2;
```

Which array?

Which spot?



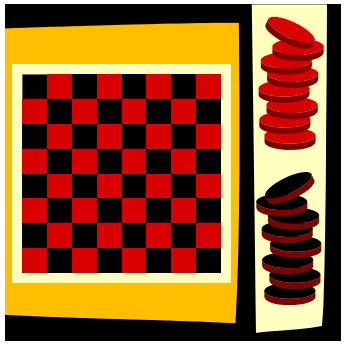
Matrices

	0	1	2	3	4
0	0	0	0	5	0
1	0	0	0	0	0
2	0	0	7	0	0
3	0	0	0	0	0
4	0	3	0	0	0

mat[2][2]=7;

mat[0][3]=5;

mat[4][1]=3



Matrices

```
for( int r = 0; r < mat.length; r++)
{
  for( int c = 0; c < mat[r].length; c++)
  {
       mat[r][c] = r*c;
  }
}
```

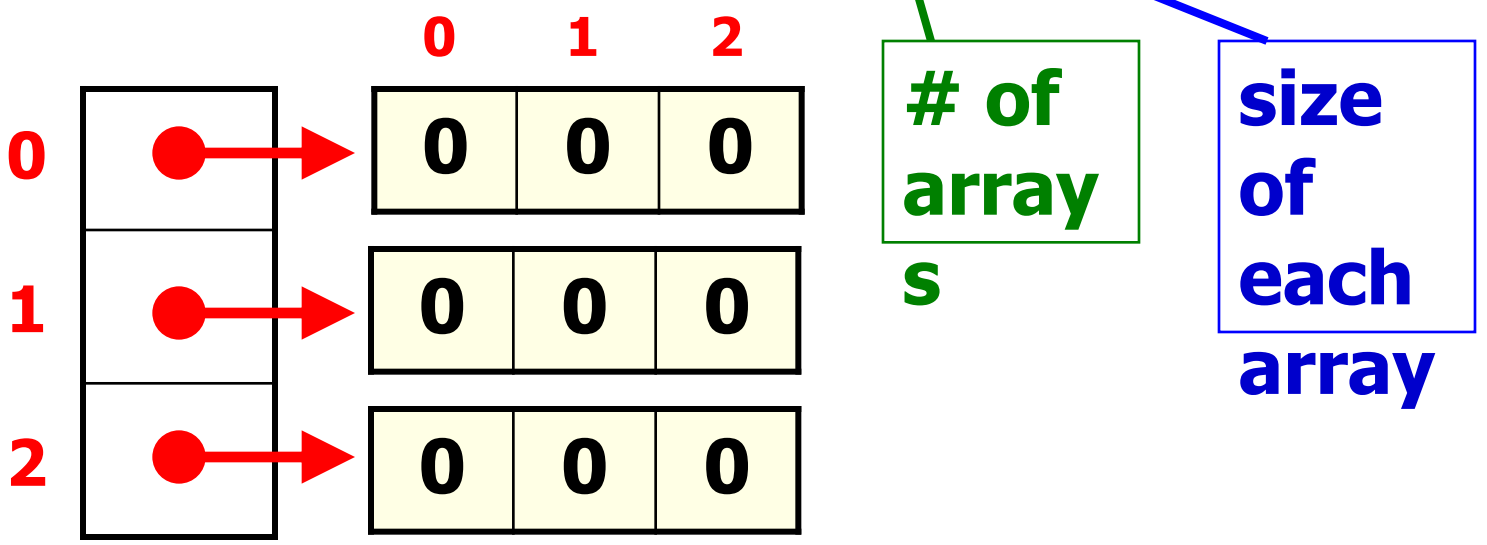
if mat was 3x3

0	0	0
0	1	2
0	2	4

Matrices

A matrix is an array of arrays.

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



Matrices

```
int[][] mat = {{5,7},{5,3,4,6},{0,8,9}};

for( int[] row : mat )
{
  for( int num : row )
  {
    System.out.print( num + " ");
  }
  System.out.println();
}
```

OUTPUT

```
5 7
5 3 4 6
0 8 9
```

```
public int countWhitePixels()
{
  int count = 0;
  for( int[]   row   :   pixelValues  )
  {
     for(  int    item   :   row  )
     {
        if( item == 255 )
           count++;
     }
  }
  return count;
}
```

2012 Question 4 part A

2012 Question 4 - part B

```
public void processImage()
{
  for( int r = 0; r < pixelValues.length-2; r++)
  {
    for( int c = 0; c < pixelValues[0].length-2; c++ )
    {
      pixelValues[r][c] -= pixelValues[r+2][c+2];
      if( pixelValues[r][c] < BLACK )
        pixelValues[r][c] = BLACK;
    }
  }
}
```

A+ Computer Science

AP REVIEW

2012 FR QUESTIONS

Provided by A+ Computer Science

Visit us at

www.apluscompsci.com

Full Curriculum Solutions

M/C Review Question Banks

Live Programming Problems

Tons of great content!

www.facebook.com/APlusComputerScience

Free Response

-Read all 4 questions before writing anything

-answer the easiest question 1st

-most times question 1 is the easiest

-see if part B calls part A and so on

-many times part C consists of A and B calls

-write something on every question

-write legibly / use PENCIL!!!!!!!!!!

-keep track of your time

Free Response

-When writing methods

  -use parameter types and names as provided

  -do not redefine the parameters listed

  -do not redefine the methods provided

  -return from all return methods

  -return correct data type from return methods

Free Response

-When writing a class or methods for a class

  -know which methods you have

  -know which instance variables you have

  -check for public/private on methods/variables

  -return from all return methods

  -return correct data type from return methods

Free Response

-When extending a class

  -know which methods the parent contains

  -have the original class where you can see it

  -make sure you have super calls

  -check for public/private on methods/variables

  -make super calls in sub class methods as needed

Free Response

-When extending abstract / implementing interface

  -know which methods the parent contains

  -have the original class where you can see it

  -make sure you have super calls

  -check for public/private on methods/variables

  -make super calls in sub class methods as needed

  -implement all abstract methods in sub class

Free Response Topics

ArrayList of References / Objects
– get,set,remove,add,size – levels of abstraction

Matrix / 2 D Array
– nested loops, GridWorld (grid)

GridWorld or Make a Class
– location, actor, bug, critter, grid, super, abstract

String / Array Question
– find biggest, find smallest, etc.