1. Write your name clearly on the front of the Scantron sheet.
2. ON THE BACK of the Scantron, print your last name first AND BUBBLE IN THE BOXES.
3. ON THE BACK of the Scantron, Print your UAlbany 9 decimal digit ID number AND BUBBLE IN THE BOXES. IF YOU DON'T KNOW (or are unsure of) your ID number, take a break during the exam and have the prof. or TA look up and write the number for you.
4. Don't bother with gender, birthday, etc. But if you omit writing and bubbling in YOUR NAME AND YOUR 9-digit ID, we will do it AND MAY DEDUCT 5 points from your exam score!

4. RIP OFF this front page of this bundle of questions. WRITE YOUR NAME AND NET-ID CLEARLY ON THE BACK, and use the back to write the Java programming exam assignment.

5. PLEASE TAKE HOME the question bundle, but turn in 2 paper sheets: The scantron AND the paper where you wrote your programming question solution!

6. Continue your Java coding onto the space below if you need to. Also, fresh paper is available on request.

(20-early 21st Century electronic) computers follow the instructions coded in their programs:
   (A.) Exactly in the order determined by the code and the rules of Java except that logic errors are detected at compile time, so wrong results cannot be printed.
   (B.) Even if some code has no purpose, all the code is followed in the order determined by the program code and the rules of Java.
   (C.) In the order that makes sense to solve the problem correctly.
   (D.) Randomly.

Is is possible for somebody to write a computer program so that, when you run it, exactly which instructions the computer runs does depend on what digital image files or other data you input, in addition to the exact order and composition of the lines of Java code in the program?

   (A.) Yes, this is quite possible.
   (B.) No, this is not possible ever.
import java.util.Scanner;

public class ____________ {
    public static void main(String[]a) {
        //Write name&NetID so we can record your programming grade!
        System.out.println("Author___________________________");
        System.out.println("NetID:___________________");
        int howManyNegOrZero, howManyLargests, largest;
        //REMEMBER TO INITIALIZE ALL
        //VARIABLES! (You may use shorter names as long as they are not misleading.

        for( int i = 0; i < 10; i = i + 1 )
        {
            //continue your work on page 1 if necessary.
            System.out.println("There were " + howManyNegOrZero + " 0s or negs.");
            if( ) {System.out.println("The largest pos. > 0 was " + largest);} else { System.out.println("None.");}
            for(int i=0; i<10; i++) {System.out.println( A[i] );}
            System.out.println("The largest >0 was input " + howManyLargests + " times");
        } //end of the body of method main.
    } //end of class definition

    //2 of 8 pages
When the computer runs the code

```java
Picture picRef = new Picture(FileChooser.pickAFile());
Pixel pxA = picRef.getPixel(0,0);
pxA.setColor( picRef.getPixel(1,1).getColor() );
```

which of the four actions A., B., C., D. happens first?

(A.) The RGB intensities stored in the Pixel at location (1,1) may change.
(B). The RGB intensities stored in the Pixel at location (0,0) may change.
(C.) A Pixel is gotten from the Picture from location (1,1).
(D.) A Color is gotten from a Pixel for the purpose of copying its intensity values to some other Pixel.

When the computer runs the same code above, which of the four actions cannot ever happen?

(A.) The RGB intensities stored in the Pixel at location (1,1) may change.
(B.) The RGB intensities stored in the Pixel at location (0,0) may change.
(C.) A Pixel is gotten from the Picture from location (1,1).
(D.) A Color is gotten from a Pixel for the purpose of copying its intensity values to some other Pixel.

In Project 6, you declared a House (reference) variable with code like `House myHouse;` and a simple Java `int` variable with code like `int maxVotes;` Which of the 4 choices is most accurate about Java?

(A.) Both variables store addresses. They store the location in computer memory where bits representing the integer's actual value are, and where in memory is the House object.
(B.) Both variables store values or objects. The int variable stores bits representing the integer's actual value and the House variable stores the actual House object.
(C.) There is a difference: The int variable stores the bits representing the integer's actual value but the House variable stores the address of the House object.
(D.) There is a difference: The int variable stores the address or location of where in computer memory are the bits representing the integer, but the House variable stores the actual House object.

An array of `int` variables referred to by `A` starts out so when the code

```java
for( int i = 0; i < A.length; i++ )
{
    System.out.print(" " + A[i]);
}
```

is run, the computer prints

```
10 20 30 40
```

After printing this, suppose the computer performed the code:

```java
A[ 1 ] = A[ 0 ]; Hint: Name of variable = value to write on it;
```

After these 3 lines of code, suppose the computer performed again

```java
for( int i = 0; i < A.length; i++ ) {System.out.print(" " + A[i]);}
```

What is printed this time? It is really wise to draw the array and actually trace the erasing and rewriting on paper because people often make mistakes when they try problems like this in their heads.

```
a. 10 10 20 30
d. 10 10 10 10
b. 40 30 20 10
e. 40 40 40 40
c. 20 30 40 40
```
** Method of a future Project 6 House class. When called, adds the
* given Picture to the House's art collection at the far left
* position [0] and discards the rightmost picture [5] to make room.
* The Pictures originally in positions 0, 1, 2, 3 and 4 are
* "shoved" one spot to the right. The likeCount values are kept
* consistent and the given Picture starts out with 0 likes.
* @param pict Specifies the given Picture to be added.
*/

```java
public void shove(Picture pict)
{  
    int i;
    i = 0;
    while( i < 5 )
    {  
        this.pictArr[i+1]=this.pictArr[i];
        this.likeCount[i+1]=this.likeCount[i];
        i = i + 1;
    }
    this.pictArr[0] = pict;
    this.likeCount[0] = 0;
}
```

Which of the two codes above correctly implement the method described by the javadoc comment?

Suppose the user successfully picks the three images One.jpg, Two.jpg and Three.jpg in that
order when he or she runs the program BELOW. Which two images will he or she actually see?

```java
public class Sprl4FinalApp
{
    public static void main(String[]a)
    {
        FileChooser.pickMediaPath();
        Picture Sally;
        Picture Charlie;
        Picture Bill;
        Charlie = new Picture( FileChooser.pickAFile() );
        Sally = new Picture( FileChooser.pickAFile() );
        Bill = new Picture( FileChooser.pickAFile() );
        Sally = Bill;
        Sally.explore();
        Charlie.explore();
    }
}
```
Compare the Java expressions $2*i+1$ and $2*(i+1)$ and $(2*i)+1$. What is most accurate, according to Java's syntax rule that multiplication (like in math) has higher precedence than addition?

(A) $2*i+1$ and $2*(i+1)$ mean the same computation and $(2*i)+1$ is different.
(B) $2*i+1$ and $(2*i)+1$ mean the same computation and $2*(i+1)$ is different.
(C) $(2*i)+1$ and $2*(i+1)$ mean the same computation and $2*i+1$ is different.
(D) $(2*i)+1$ and $2*(i+1)$ and $2*i+1$ all mean the same computation.
(E) $(2*i)+1$ and $2*(i+1)$ and $2*i+1$ all mean three different computations.

Which explanation of the Java expression $A[2*i+1]$ best explains how the value and the location of a Java array element are denoted in Java syntax? Suppose the array was made (instantiated) by `int[] A = new int[10];`

(A) In $A[2*i+1]$, $A$ is the array name, $i$ is the element location, and $2*i+1$ is the element's value.
(B) In $A[2*i+1]$, $i$ is the array name, $A$ is the location, and $2*i+1$ is the element's value.
(C) In $A[2*i+1]$, $A$ is the array name, $i$ is the element location, and $A[2*i+1]$ is the element's value.
(D) In $A[2*i+1]$, $A$ is the array name, $2*i+1$ is the element location, and $A[2*i+1]$ is the element's value.
(E) In $A[2*i+1]$, $A$ is the array name, $A[i]$ is the element location, and $A[2*i+1]$ is the element's value.

Which is the (one) correct flowchart for the logical meaning of Java's `while` statement, which has the syntax

```java
while( <TEST> ) {
  <BODY STATEMENTS>
}
```

(A) 
(B) 
(C) 

What is the last number value that the computer copies into variable $k$ when the computer runs the code below? In other words, what does it print?

```java
int k;
k = 100;
while( k > 0 )
{
  k = k - 1;
}
System.out.println( k );
```

(A) 100
(B) 99
(C) -1
(D) 0
(E) 1
Tracing with an array. The boxes on the right show what the code below before the while did, plus a little of what you must figure out. Trace rest of the using the boxes underneath to carefully determine everything the computer would have in the array elements and in variable k after each computation step, one by one. Then, select the answer that shows all the array element values at the time when all the code finishes. Use the k boxes and the blank array element boxes as if time progresses from top to bottom.

```java
int arr[] = new int[5];
for(int i=0; i<5; i++)
{   arr[i] = 2*i+1; }
int k;
k = 0;
while( k < arr.length – 1 )
{
   arr[k] = arr[k] + arr[k+1];
   k = k + 1;
}
```

(A.) 4 3 5 7 9
(B.) 4 8 5 7 9
(C.) 4 8 12 7 9
(D.) 4 8 12 16 9
(E.) 1 8 12 16 20
One lesson from Project 06 was that an extra variable for temporary use is needed in code that swaps the values of two variable using assignments. Which code below swaps the values in int X and Y?

(A.) int temp; X = Y;  temp=X;  Y=temp;
(B.) int temp; X=temp;  Y=X;  temp=Y;
(C.) int temp; temp=X;  Y=X;  X=temp;
(D.) int temp; temp=X;  Y=temp;
(E.) int temp; temp=X;  Y=X;  X=temp;

Hints: After a good guess, TEST IT WITH AN EXAMPLE LIKE WHAT HAPPENS AFTER SAY X = 1; and Y = 2; Also, it's OK with the Java compiler to put 4 statements on one line, even thought it's terrible style for people.

Java's return operation:

(A.) Might or might not return a value but it sometimes makes the computer return to the beginning of the called method's loop.
(B.) Sometimes makes the computer return to the spot where the method was called but sometimes it makes the computer return to the top of a loop, the origin of a Picture, etc.
(C.) Always (except perhaps when main returns) makes the computer return to the spot where the method was called. You can choose whether or not to return a value.
(D.) Always (except perhaps when main returns) makes the computer return to the spot where the method was called. You MUST omit the return value when returning from a constructor or void method and you MUST provide a return value otherwise.

public class ArQ1 {
    public static void main(String[]a) {
        int A[] = new int[10];
        for( int i = 0; i < 10; i=i+1 )
        {
            A[i] = i;
        }
        for( int i = 0; i < 10; i=i+1 )
        {
            System.out.print( A[i] + " ");
        }
        System.out.println();
    }
}

What does the above program print when it runs?
(A.) 0 1 2 3 4 5 6 7 8 9
(B.) 1 2 3 4 5 6 7 8 9 0
(C.) 9 8 7 6 5 4 3 2 1 0
(D.) 9 8 7 6 5 5 6 7 8 9
(E.) 8 7 6 5 4 4 5 6 7 8

public class ArQ2 {
    public static void main(String[]a) {
        int A[] = new int[10];
        for( int i = 0; i < 10; i=i+1 )
        {
            A[i] = i;
        }
        for( int i = 0; i < 10; i=i+1 )
        {
        }
        for( int i = 0; i < 10; i=i+1 )
        {
            System.out.print( A[i] + " ");
        }
        System.out.println();
    }
}

What does the above program print when it runs?
(A.) 0 1 2 3 4 5 6 7 8 9
(B.) 1 2 3 4 5 6 7 8 9 0
(C.) 9 8 7 6 5 4 3 2 1 0
(D.) 9 8 7 6 5 5 6 7 8 9
(E.) 8 7 6 5 4 4 5 6 7 8
Program writing (30 points) Write your work on page 2 or on a blank paper if you mess up page 2.

Write a complete Java application that does the following, and prints as explained:

1. (0.3 pt.) Name your app anything you want.
2. (0.3 pt.) Make a length 10 array and an array reference variable to refer to it.
   (Reference: \texttt{int[]} \texttt{A} = \texttt{new int[ 10 ];})
   (0.4 pt.) Construct or instantiate a \texttt{java.util.Scanner} from \texttt{System.in} so you can use it to get input. Reference:
   \texttt{java.util.Scanner \texttt{sc} = new java.util.Scanner(System.in);} or save clutter with \texttt{import java.util.Scanner;} at the beginning of your code file.)
3. (5 pt.) Simply input 10 \texttt{int}s into the array.) Code a loop (use loop variable \texttt{i}, where \texttt{i} will range over 0, 1, 2, ..., 9 in that order, and the following is done during each repetition of the body: Input one integer and store it in the element of array \texttt{A} whose index is the value of \texttt{i}.
   (Reference: (1) "=" means copy or store, not equals. (2) \texttt{A[ some expression ]} refers to one array element variable. (3) \texttt{sc.nextInt();} inputs and returns one integer value.)
4. Demonstrate programming with one or more NON-NESTED loops to compute the values to be printed AND change the data in some array elements described below. Then \textbf{FINISH OUR PRINTING CODE} to make your program print what is specified, on separate lines, in the \textit{exact a,b,c,d order below}: (TIP: \textbf{REVIEW} your work to ensure that your code written ABOVE our printing code computes and stores the correct data to be printed.)
   a) (6 pts.) HOW MANY of the \texttt{int}s in the array are negative or zero (\texttt{A[i] \textless 0})
   b) (6 pts.) The LARGEST among the strictly positive ( \texttt{> 0} ) only \texttt{int}s in array \texttt{A} if any; if ALL the \texttt{int}s are negative or 0, print \texttt{None} instead of any number.
   c) (6 pts.) (c1) Go through the array again and SUBTRACT 1 from every copy of the largest among the strictly positive array elements. (c2) Print all the array element VALUES AFTER any or all copies of the largest (among the strictly positive) were reduced by 1, in the order of their indices 0, 1, ..., 9
   d) (6 pts.) HOW MANY TIMES did the largest appear among the \textbf{original ten int}s stored in the array. (HINT: Decide for yourself the order in which the computer should do what it should do.)

Java reference info:
\begin{verbatim}
if ( TEST ) { ..(body). } and
if ( TEST ) { ..(body) } else { ..(body) } make a decision once.
while( TEST ) { ..(body).. } (Remember to use loop variables! and code INIT and UPDATE when you use whiles!) and
for( INIT ; TEST ; UPDATE ) { ..(body). } make decisions repeatedly to make the computer repeat the loop body code.
\end{verbatim}

Examples of TESTs: (A "TEST" is a boolean expression.)
(1) \texttt{i < 10} (2) \texttt{A[ i ] > biggie} (3) \texttt{A[i] > A[j]}

How to decide which ones to use, and how to use them, in a programming questions, cannot be summarized in a cheat sheet! That's for you to learn by practice and study.