AP Computer Science A Name_____

Teacher_____

Period_____

Packet for the weeks of March 16-April 3, 2020

Week of March 16th

Unit 8 Review- AP Style Questions

Unit 8 Vocab Review

Assignment 7 – Battleship

Week of March 23rd

Review Unit Introduction

Diagnostic Exam

Week of March 30th

Programming Fundamentals and Review Questions

Data Structures and Review Questions

Logic and Review Questions

Unit 8 Review Questions - Instructions

Test your knowledge with some AP-style practice!

Select the correct answer for each question on the quiz.

Remember: **some questions require two answers**. You must select **both** correct answers to get full credit for the question. Selecting 1, 3 or 4 answers will mark the question as incorrect.

Flag this Question

Question 1 1 pts

Consider the following method.

```
public static int operation(int[][] mat)
        {
          int currentVal = mat[0][0];
          int result = 0;


  for(int j=0; j < mat.length; j++);
  {
    for(int k=0; k < mat[j].length; k++)
    {
      /* missing code */
    }
  }


  return result;
}
```

Which of the following should replace /* missing code */ so that the method returns the index of the row which contains the smallest value in the two-dimensional array?

```
if (mat[j][k] < currentVal)
{
  currentVal = mat[j][k];
  result = k;
}
```

```
if (mat[j][k] > currentVal)
{
  currentVal = mat[j][k];
  result = j;
}
```

```
if (mat[j][k] < currentVal)
{
  currentVal = j;
  result = mat[j][k];
}
```

```
if (mat[j][k] > currentVal)
{
  currentVal = k;
  result = mat[j][k];
}
```

```
if (mat[j][k] < currentVal)
{
  currentVal = mat[j][k];
  result = j;
```

```
}
```

Flag this Question

Question 21 pts

Consider the following code segment.

```
int[][] mat = new int[4][6];
        for (int r = 0; r < 4; r++)
        {
          for (int c = 0; c < 6; c++)
          {
            if(c < 3 - r || c > 3 + r)
            {
              mat[r][c] = 1;
            }
            else
            {
              mat[r][c] = 0;
            }
          }
        }
```

Which of the following represents mat after this code segment is executed?

1	1	1	0	1	1
1	1	0	0	0	1
1	0	0	0	0	0
0	0	0	0	0	0

1	1	1	1	1	1

1	1	1	0	1	1
1	1	0	0	0	1
1	0	0	0	0	0

1	1	1	1	1	1
1	1	0	0	1	1
1	0	0	0	0	1
0	0	0	0	0	0

1	1	0	0	1	1
1	0	0	0	0	1
0	0	0	0	0	0
0	0	0	0	0	0

1	1	0	1	1	1
1	0	0	0	1	1
0	0	0	0	0	1
0	0	0	0	0	0

Flag this Question

Question 3 1 pts

Consider the following method, which is intended to return an array which contains the maximum elements in each of the rows of a 2-dimensional array.

```
        /** @param  mat a 2-dimensional array
 *  @return an array which contains the maximum elements of each row in mat.
 */
public double[] maxRows(double[][] mat)
```

```
{
  double[] maxes = new double[mat.length];

  for (int k = 0; k < mat.length; k++)
  {
    double localMax = mat[k][0];

    for(double num : mat[k])
             {
      /* missing code */
    }

    maxes[k] = localMax;
  }

return maxes;
}
```

Which of the following could be used to replace /* missing code */ so that maxRows will work as intended?

```
  if (num > localMax)
    localMax = mat[k][num];
```

```
  if (mat[num] > localMax)
    localMax = mat[num];
```

```
  if (mat[k][num] > maxes[k])
    localMax = mat[k][num];
```

```
  if (mat[k][num] > localMax)
    localMax = num;
```

```
if (num > localMax)
localMax = num;
```

Flag this Question

Question 4 1 pts

Consider the following method.

```
public String mystery(String s1, String s2)
{
  String output = “”;
  int len;

  if (s1.length() < s2.length())
    len = s1.length();
  else
    len = s2.length();

  for (int k = 0; k < len; k++)
  {
    output += s1.substring(k, k+1);
    output += s2.substring(k, k+1);
  )

  return output;
}
```

What is returned as a result of the call mystery(“Alice”, “Bob”)?

BAolbi

ABloib

Nothing is returned because an IndexOutOfBoundsException is thrown.

AliceBob

ABloibc

Unit 8 Vocabulary Review

Match the vocabulary word with the correct definition.

Arrays	When you need to store a table of data - creates an array of arrays. Array with both rows and columns.	Variable that holds many pieces of data at the same time. Store one type of data, indexed from 0 to length - 1, and used to store large amounts of data.
Two dimensional arrays	Means that the row comes first.	Row - major

Assignment 7 - Battleship Instructions

In this assignment, you will create a simple class for playing the game battleship. Battleship is played on a square grid of 10 rows and 10 columns, which you will represent using a 2-d array in a class named Board. A separate class, Battleship, will allow you to try playing your game and see whether the methods work as planned.

To play battleship, 2 players secretly add ships of various lengths to their individual boards. A ship may be placed horizontally or vertically on the board. In your program, you will represent blank squares on the board using the character '-', and squares where ships have been added with the character 'b'. For example, the board below contains 3 boats, 2 of length 5 and one of length 3:

```
- - - - - - - m - -
- x x x - - m x - -
- - - - - - - x - -
```

```
- - - m - - - b - -

- - - - - - - b - -

- m - - - - - b - -

- - - - m - - - - -

- - b b b b b - m -

- - - - - - - - - -

- - - - - m - - - -
```

Players then take turns to try and shoot the ships on each other's boards by choosing a row and column reference for each attempted shot. When a player chooses a square containing a ship, that square is marked as a "hit": in your program, this will be done using the character 'x'. When the player "misses" by choosing a blank square this will be marked with the character 'm' in your program. The game is over when one player has shot every square containing part of a ship on their opponent's board. The board below is from a game in progress. One ship has already been completely destroyed and 2 squares of one of the other ships have been shot.

```
- - - - - - - m - -

- x x x - - m x - -

- - - - - - - x - -

- - - m - - - b - -

- - - - - - - b - -

- m - - - - - b - -

- - - - m - - - - -

- - b b b b b - m -

- - - - - - - - - -

- - - - - m - - - -
```

The specification for the Board class is as follows:

Variables

char[][] squares - An array which represents the board on which a player places their battleship and records shots.

Constructors

Board() - This constructor initializes the squares array with every value set to '-', which is the character used to represent a blank square.

Methods

- **public String toString()** - Returns a multi-line representation of the board by printing each character in squares with spaces after each character and a new line for each row.
- **public boolean addShip(int row, int col, int len, boolean horizontal)** - Attempts to add a ship of length len to the grid, starting at the row and column specified and proceeding either rightwards (if horizontal is true), or downwards (if vertical is true). If the ship can be placed in the place specified, each square making up the ship should be set to 'b', and the method should return true. A ship may not be placed if it would go off the grid, or would intersect another ship on the grid. If the ship cannot be placed, no values in squares should be changed and the method should return false.
- **public int shoot(int row, int col)** - If row and col specify a square which is out of bounds, the method should return -1. If the square at the specified row and column contains '-' (i.e. is blank), the square should be changed to 'm' to signify a miss, and the method should return 0. If the square contains 'b' (i.e. a battleship which hasn't been hit yet) this square should be changed to an 'x' to signify a hit, and the method should return 1. If the square contains either 'x' or 'm' the method should return 2 (these are squares which have already been "shot").
- **public boolean gameOver()** - If the character 'b' does not appear at any location in squares, then there are no unsunk ships remaining on the board, so return true to indicate that the game is over. Otherwise return false.
- **public boolean foundShip(int len)** - Search the board for any possible ships of length len. If there are exactly len consecutive squares (either horizontal or vertical) containing a 'b' character somewhere in the grid, then return true, otherwise, return false.

When you have created your Board class, download the Battleship.java (Links to an external site.) class to the same folder. When you run the main method in this class, you will be prompted to add ships to the board, then to attempt to shoot these. See the sample run below for an example.

Sample Run

```
Welcome to Battleship!


- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -
```

```
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -



Add a new ship? (y/n)
 n

You need a ship of length 3 to play!

Add a new ship? (y/n)
 y
Starting in which row?
 2
Starting in which column?
 3
How long?
 3
Horizontal (h) or vertical (v)?
 h

New ship added!

- - - - - - - - - -
- - - - - - - - - -
- - - b b b - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
```

```
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -



Add a new ship? (y/n)
 n

Ok, let's play!

Input row
 5
Input column
 6

Miss!

- - - - - - - - - -
- - - - - - - - - -
- - - b b b - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - m - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -



Input row
```

```
 2
Input column
 3


Hit!


- - - - - - - - - -
- - - - - - - - - -
- - - x b b - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - m - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - - - - -



Input row
 2
Input column
 4


Hit!


- - - - - - - - - -
- - - - - - - - - -
- - - x x b - - - -
- - - - - - - - - -
- - - - - - - - - -
- - - - - - m - - -
```

```
- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -



Input row

 2

Input column

 5


Hit!


- - - - - - - - - -

- - - - - - - - - -

- - - x x x - - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - m - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -

- - - - - - - - - -



Game over!
```

You should run the game several times and try different inputs - this uses the methods from your Board class, so these will all need to produce the correct results and outputs if the program is to work as desired. You can also add, edit, or isolate code to help you with testing specific features.

When you are ready, paste your entire Board.java class from start to finish in the box below, click run to test the output, and click submit when you are satisfied with your results.

Milestones

As you work on this assignment, you can use the milestones below to inform your development process:

Milestone 1: Write the Board class header, declare the squares variable and write the single constructor required.

Milestone 2: Write and test the toString and gameOver methods.

Milestone 3: Write and test the addShip method.

Milestone 4: Write and test the shoot method. Run the full Battleship program to test all the features.

Review Unit Introduction

Unit 9 is all about preparing for the AP Exam. Our focus will be on bringing together everything we've learned this year and applying it to the types of questions you will see on the exam. We'll also discuss test-taking tips and strategies.

This unit is structured a bit differently than previous units. Instead of a set of numbered, sequential lessons, you'll see to the left there are topics: Programming Fundamentals, Data Structures, etc. These topics are specific to the AP Exam. Also, rather than releasing a unit over time, all of the topics are available to you now.

Start this unit by first taking the diagnostic exam. This is a great way to assess your individual strengths and weaknesses before you begin your dedicated exam prep.

Then, go through each topic area. The results on your diagnostic exam will help inform which topics to focus on. Spend time watching the review videos and answering the related questions. Go back and work through previous lessons and reference materials.

Remember, Piazza is a great place to ask questions, post comments and share study tips with other students. If you haven't tried Piazza yet this year, now's a great time to start. To get to Piazza, click the "Student Forum" button in the sidebar on the left.

Diagnostic Exam on a separate document

Programming Fundamentals PowerPoint slides

Programming Fundamentals Review Question

Question 1 1 pts

The code below is intended to search through an ArrayList of Strings and remove any String not starting with a letter.

Which of the following will work as intended in all possible scenarios?

I.

```
for (int i = 0; i < list.size(); i++) {
    char let = list.get(i).charAt(0);
    if (!((let >= 'a' && let <= 'z') || (let >= 'A' && let <= 'Z'))) {
        list.remove(i);
    }
}
```

II.

```
for (String s : list) {
    char let = s.charAt(0);
    if (!((let >= 'a' && let <= 'z') || (let >= 'A' && let <= 'Z'))) {
        list.remove(s);
    }
}
```

III.

```
for (int i = list.size() - 1; i >= 0; i--) {
    char let = list.get(i).charAt(0);
    if (!((let >= 'a' && let <= 'z') || (let >= 'A' && let <= 'Z'))) {
        list.remove(i);
    }
}
```

II only

III only

I and III only

I only

I, II and III

Data Structures - Review Questions

Question 1 1 pts

Consider the following method intended to count the number of times the value `v` is stored in a 2-D array.

```
    public static int countIt(int g[][], int v)
    {
        int co = 0;

        /* missing code */

        return co;
    }
```

Which of the following could be used to replace `/* missing code */` so the method works as intended?

```
for (int r = 0; r < g.length; r++)
{
    for (int c = 0; c < g[0].length; c++)
    {
        if (g[r][c] == v)
            co++;
    }
}
```

```
if (g[0][grid[0].length - 1] == v) co++;
```

```
for (int c = 0; c < g[0].length; c++)
{
    if (g[r][c] == v)
        co++;
}
```

```
co++;
```

```
if (g[0][0] == v) co++;
```

Question 2 1 pts

Assume a 2-D array of integer values has been declared as follows and has been initialized.

```
int[][] grid = new int[3][5];
```

Which of the following correctly swaps the first and second rows of the 2-D array?

```
    for (int i = 0; i < grid[0].length; i++) {
        grid[0][i] = grid[1][i];
        grid[1][i] = grid[0][i];
    }
```

```
    for (int i = 0; i < grid.length; i++) {
        int temp = grid[i][0];
        grid[i][0] = grid[i][1];
        grid[i][1] = temp;
    }
```

```
    for (int i = 0; i < grid[0].length; i++) {
        int temp = grid[1][i];
        grid[1][i] = grid[2][i];
        grid[2][i] = temp;
    }
```

```
    for (int i = 0; i < grid[0].length; i++) {
        int temp = grid[0][i];
        grid[0][i] = grid[1][i];
        grid[1][i] = temp;
    }
```

```
    for (int i = 0; i < grid.length; i++) {
        int temp = grid[i][1];
        grid[i][1] = grid[i][2];
        grid[i][2] = temp;
    }
```

Logic - Review Questions

Question 1 **1 pts**

Consider the following code.

```
int x = /* some int value */;
int y = /* some int value */;
boolean answer = Math.abs(x/y) > (x/y);
```

Under what conditions is `answer` true?

When x and y have different signs.

When x and y are equal.

When both x and y are positive.

When both x and y are negative.

When x and y have the same signs.

Question 2 1 pts

Consider the following code intended to test if a coupon can be applied to a pizza order. The coupon applies only if the pizza is large and has two or more toppings.

```
String size = /* code not shown */;
int numToppings = /* code not shown */;
boolean canUseCoupon;
if ( /* missing code */ ) {
    canUseCoupon = true;
} else {
    canUseCoupon = false;
}
```

What code can replace /* missing code */ so that the correct result is stored in `canUseCoupon`

```
!(size.equals("large") || numToppings >= 2)
```

```
size.equals("large") || numToppings >= 2
```

```
size.equals("large") && numToppings > 2
```

```
size.equals("large") && numToppings >= 2
```

```
size.equals("large") || numToppings > 2
```

Question 3 1 pts

Consider the following line of code.

```
(!(a == b && a != b))
```

When is the condition true?

When a is not equal to b.

When a is greater than b.

The statement is never true.

When a is equal to b.

For all real numbers a and b.

Question 4 1 pts

'Consider the following method `getPrice` that is used in a class to represent a pizza. The method `getPrice` is based on the following:

```
public class Pizza {

    private String size;
    private int numToppings;

    /* other code not shown */

    public double getPrice() {
        /* missing code */
    }
}
```

The pricing is based on the following assumptions.

- small pizza: $6.99

- medium: $8.99
- large $10.99

The first two toppings are free, and any additional toppings are $1.99 each.

Which of the following could be used in place of `/* missing code */` so that the method works as intended?

I.

```
    double price = 6.99;

    if (size.equals("medium")) {
        price = 8.99;
    }

    if (size.equals("large")) {
        price = 10.99;
    }

    if (numToppings > 2) {
        price = price + (numToppings - 2) * 1.99;
    }

    return price;
```

II.

```
    double price = 6.99;

    if (numToppings > 2) {
        price += (numToppings - 2) * 1.99;
    }

    if (size.equals("medium")) {
        price += 2;
    }

    if (size.equals("large")) {
        price += 4;
    }

    return price;
```

III.

```
    double price = 6.99;

    if (size.equals("medium") && numToppings > 2) {
        price = 8.99 + (numToppings - 2) * 1.99;
    }

    if (size.equals("medium")) {
        price = 8.99;
    }

    if (size.equals("large") && numToppings > 2) {
        price = 10.99 + (numToppings - 2) * 1.99;
    }

    if (size.equals("large")) {
        price = 10.99;
    }

    return price;
```

I and II only

III only

II and III only

I only

II only

Question 5 1 pts

Consider the following code:

```
    boolean flag = true;
    for (int i = 0; i < 100; i++) {
         if (/* missing code */)
              flag = false;
    }
```

It is intended to test if the values in the array `vals` are sorted in ascending order.

What should replace `/* missing code */` so that it works as intended?

```
vals [i] < vals [i + 1]
```

```
vals [i] <= vals [i - 1]
```

```
vals [i] > vals [i + 1]
```

```
vals [i] <= vals [i + 1]
```

```
vals [i] > vals [i - 1]
```

Question 6 1 pts

Consider the following code intended to return true when `val` is not between lower and upper inclusive.

```
    /* precondition lower <= upper */
    public static boolean outside(int val, int lower, int upper) {
         if ( /* missing code */ ) {
              return true;
         } else {
              return false;
         }
    }
```

Which of the following can be used to replace `/* missing code */` so that the method works as intended?

```
val < lower || val > upper
```

```
val <= lower && val >= upper
```

```
val < lower && val > upper
```

```
val > lower && val < upper
```

```
val > lower || val < upper
```

Question 7 **1 pts**

Consider the following line of code.

```
    (!(a == b || a < b))
```

When is the condition true?

When b is greater or equal to a.

When a is greater than b.

When a is greater or equal to b.

When a is less than b.

The condition can never be true.