Computer Science 2 Practice Quiz, Arrays and Sorts Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Matching (1 point ea.)

 1) \_\_f\_\_array a) A conjunction that is true if either condition is true.

 2) \_\_j\_\_Dry run b) A type of variable is used to store someone’s name.

 3) \_\_g\_\_ count++ c) A conjunction that is true if and only if both conditions are true.

4) \_\_H\_\_ Big ‘O’ notation d) Declaring an array of ints.

5) \_i\_\_ Stability e) The speed of a Bubble sort.

6) \_c\_ && f) A variable that can store several values.

7) \_b\_ String g) Adds one to an integer variable.

1. \_a\_ || h) A method for evaluating the speed of an algorithm.

9)\_e\_ O(n2) i) When sorting, it describes how well secondary fields stay in order.

10) \_d\_int []a = new int[10]; j) Testing code or pseudo code with examples.

II. Short Answer

* 1. What are the semantics of a while loop? Get var, while var<> flag, Squiggle, get var, end
	2. When should you use a while loop? Repeat something an unknown number of times, may not occur
	3. When should you use a do while loop? Repeat an unknown number of times, at least once
	4. What is an array? A variable that can store a large group of similar information.
	5. When should you use an array? When you want to store a large group of similar information that will be used more than once. Like when storing, Sorting, tallying

1. Declare an array (Type section only) to hold the following information(2 pts ea)**TYPE**
	1. The names for 24 schools. **String[] names = new String[24];**
	2. The ages of 17 students. **int [] ages = new int[17];**
	3. The Votes for 14 candidates **int [] votes = new int[14];**
	4. The temperatures for 100 temperature readings **int [] temps = new int[100];**
2. Use the array declared in 3b) write the section of code to get the 17 ages from the user.

for (int count = 0; count <17; count++)

{

 SOP(“Please enter your name”);

 names[count] = input.nextLine();

}

Sorts (fill in the chart for the following sorts): 5 points each.

26) **Bubble**

 Speed\_\_\_O(n2)\_\_

 Stability \_\_\_Yes\_\_\_\_\_\_\_\_\_\_

 How it works \_\_Check- switch\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Show the following after the 4th pass (Put your answer in the chart provided)

 Low High

 23 15 34 2 90 14 8 105 27 6 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 23 | 2 | 34 | 14 | 8 | 90 | 27 | 6 | 1 | 105 |
| 15 | 2 | 23 | 14 | 8 | 34 | 27 | 6 | 1 | 90 | 105 |
| 2 | 15 | 14 | 8 | 23 | 27 | 6 | 1 | 34 | 90 | 105 |
| 1 | 14 | 8 | 15 | 23 | 6 | 1 | 27 | 34 | 90 | 105 |

26) **Selection**

 Speed\_O(n2)\_\_\_\_

 Stability \_\_\_Not stable\_\_\_\_\_\_\_\_\_\_

 How it works \_\_\_Check-Mark-Switch\_\_\_\_\_\_\_\_\_\_\_\_\_

 Show the following after the 4th pass (Put your answer in the chart provided)

 Low High

 23 15 34 2 90 14 8 105 27 6 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 15 | 34 | 2 | 90 | 14 | 8 | 105 | 27 | 6 | 23 |
| 1 | 2 | 34 | 15 | 90 | 14 | 8 | 105 | 27 | 6 | 23 |
| 1 | 2 | 6 | 15 | 90 | 14 | 8 | 105 | 27 | 34 | 23 |
| 1 | 2 | 6 | 14 | 90 | 15 | 8 | 105 | 27 | 34 | 23 |

26) **Insertion**

 Speed\_O(n2)\_\_\_

 Stability \_\_Yes\_\_\_\_\_\_\_\_\_\_\_

 How it works \_\_Dummy – slide - back\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 Show the following after the 4th pass (Put your answer in the chart provided)

 Low High

 23 15 34 2 90 14 8 105 27 6 1

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 | 23 | 34 | 2 | 90 | 14 | 8 | 105 | 27 | 6 | 1 |
| 15 | 23 | 34 | 2 | 90 | 14 | 8 | 105 | 27 | 6 | 1 |
| 2 | 15 | 23 | 34 | 90 | 14 | 8 | 105 | 27 | 6 | 1 |
| 2 | 15 | 23 | 34 | 90 | 14 | 8 | 105 | 27 | 6 | 1 |

 al

0 -2 6

1 3 16

2 8 12

3 13 6 24 16

4 18 12

Screen

-2

3

8

6

12

6

16

12

16

12

{Dry Run}

int [] al = new int[5];

for (int ready= 0; ready< 5; ready++)

al[ready]= 5\*ready - 2;

for (int uminum= 4;uminum >=3; uminum--)

al[uminum]= al[uminum - 2]+ uminum;

for (int ready= 1; ready<5;ready++)

 System.out.println(al[ready]);

for (int togethernow= 0; togethernow<4; togethernow++)

 al[togethernow]=2\*al[togethernow + 1];

al[3]=al[1];

for (int ready= 0; ready< 5; ready++)

System.out.println(al[ready]);

Num

-3

-2

1

6

13

Screen

13

6

1

-2

-2

Assume the arrays have been previously declared holding 5 integers.( 4 pts ea)

35) for (int count = 0;count<5; count++)

 num[count] = count\*count - 3;

 for (int count = 4; count>=0; count--)

 System.out.println(num[count]);

36) for (int count:= 0; count<2; count++)

Scores row col

0 0 0 0 1 2

1 6 1 0 1 2

2 6 2 0 1 2

 3 0 1 2

Screen

0

6

 scores[count]=0;

 for (int row= 0; row<4; row++)

 for (int col=0; col<3; col++)

 {

 if (row+col) % 2 == 0 then

 scores[1]= scores[1]+col

 else

 scores[2]= scores[2] + col;

 }

 for (int count= 0; count<2; count++)

 System.out.println(scores[count]);

Write the code for **one** of the following. Write down assumption you have made. You can use/modify the sort code below as needed.

1. Write the section of code that will get 10 names and show the names in alphabetical order.
2. Write a program to fill an array with 100 random integers in the range 1-50. Calculate the average and the **number of scores** that are within 5 of the average.
3. Write a program to roll a pair of six-sided dice 100 times. The program will show how often each roll occurred.

Sort code to modify as needed:

1. String [] names = new String[10];

String dummy;

for (int count = 0; count<10; count++)

{

 SOP(“Please enter a name”);

 names[count] = input.nextLine();

}

for (int pass= 0; pass<names.length -1;pass++)

 {

 for (int check= 0; check <names.length-1;check++)

 {

 if (names[check].compareTo(names[check+1])>0)

 {

 dummy= names [check];

 names [check]= names [check+1];

 names [check+1]= dummy;

 }//Of the if and the for check loop

 }

for (int count = 0; count<10; count++)

{

 SOP(names[count]);

}

 }//