

Arrays Introduction & Big Oh Notation

3.1

Name: _____

0. Look at the following array and answer these questions about it.

```
int m[]={1, 8, 3, 4, 20};  
  
int t = 3;  
  
//Print backwards  
String rev = "";  
for(int i=m.length-1; i>=0; i--)  
    rev+=m[i];  
result.setText(rev);
```

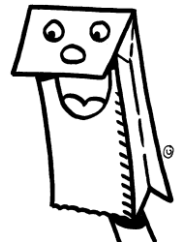
(a) Fill in the array memory diagram for 'm':

[0]	[1]	[2]	[3]	[4]

- (b) What is the value of m.length?
- (c) What is the value of m.length()?
- (d) What is in m[2]?
- (e) What is in m[t]?
- (f) What is in m[t+1]?
- (g) What is in m[t-1]?
- (h) What is in element 0?
- (i) What is the index of 8?
- (j) What is the index of 20?
- (k) What is in m[m.length]?
- (l) What is in m[m.length-1]?

1. Why are arrays useful? (Circle the most correct answer)

- (a) They allow the grouping of variables so that they can be loops through quickly.
- (b) They allows you to change the flow of the code through the program.
- (c) They allow the division of code into small sections to organize it.
- (d) They allow the programmer to make notes to themselves about the running of the code.



2. Suppose that you have a sorted array and wish to find all the repeated values:

```
int sort[]={1,2,3,5,5,6,7,8,9,11,13,13,17,20,20,21};  
  
String repeat = "";  
  
for(int i=0; i<sort.length-1; i++){  
    if(sort[i]==sort[i+1])  
        repeat+= sort[i]+" ";  
  
result.setText(repeat);  
}
```

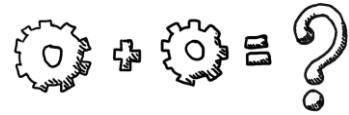
(a) What are the repeated values in the array?	
(b) What is the array type?	
(c) What is the array name?	
(d) What does the loop start at?	
(e) What does the loop end at?	

(f) Write code to print the minimum of the sorted array in a TextView named min. (handle inflation first)

(g) Write code to print the maximum of the sorted array in a TextView named max. (handle inflation first)

3. What is an algorithm? (Circle the most correct answer).

- (a) Computer code that sorts things. (c) A flowchart diagram.
(b) A series of steps to complete a task. (d) Computer code.



4. In Big Oh notation (for arrays)

- (a) what does the O stand for? (b) what does the n stand for?

5. Why do we measure algorithm speed in terms of number of operations as opposed to seconds?

6. Put these algorithm speeds in order.

(1 is fastest, 7 is slowest)

- ___ $O(n^2)$
___ $O(n)$
___ $O(n \log n)$
___ $O(2^n)$
___ $O(\log n)$
___ Constant time
___ $O(n^3)$

7. What speeds are these array algorithms?

- (a) Finding the minimum _____
(b) Printing all values _____
(c) Swapping two values _____
(d) Selection Sort _____
(e) Finding the maximum _____
(f) Finding the average _____
(g) Quicksort _____
(h) Binary search _____

8. A common coding interview question based on algorithms is:

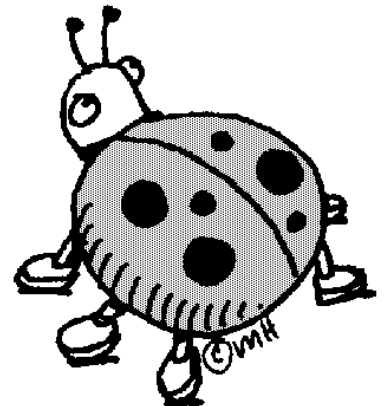
"How do you find the missing number in integer array of 1 to 100?"

(a) This code puts the values from 1-100 into an array. Fill in the blanks.

```
_____ hundred[] = new int[_____];  
for(int i = ____; i < hundred._____; i++){  
    hundred[_____] = i+1;  
}
```

(b) This code un-sorts the array from part a. Put the code in order.

```
_____ int a = (int)(Math.random()*100);  
_____ hundred[a] = hundred[b];  
_____ int swap = hundred[a];  
_____ }  
_____ hundred[b] = swap;  
_____ for(int i = 0; i < 60; i++){  
_____ int b = (int)(Math.random()*100);
```



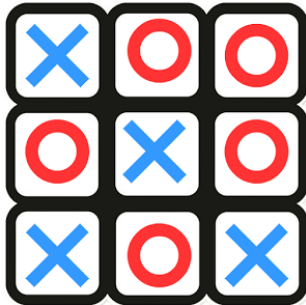
(c) Assume that after the array is sorted, a user randomly selects a number and replaces it with 0.

```
int randomPlace = (int)(Math.random()*100);  
hundred[randomPlace] = 0;
```

Write the code to find the missing number WITHOUT using randomPlace. Save it in an int named answer.
Strong hint: The sum of the numbers from 1 to 100 is 5050.

Name: _____

1. On a GridLayout, views are added row by row, from left to right. Label the order these widgets should be put into the GridLayout so that they are added correctly.



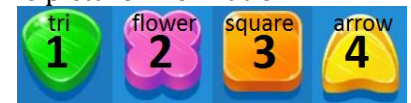
2. This game is called Scrubby Dubby. Fill in the grid information with the appropriate picture code.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#000000">

    <GridLayout
        android:layout_width="wrap_content"
        android:layout_gravity="center"
        android:layout_height="match_parent"
        android:rowCount="3"
        android:columnCount="3"
        android:id="@+id/grid">
    </GridLayout>

</LinearLayout>
```

The picture information:



The grid information:



```
public class Game extends AppCompatActivity {
    int soap[][]= {{_____, _____, _____, _____, _____},
                   {_____, _____, _____, _____, _____},
                   {_____, _____, _____, _____, _____}};

    int row = ____;
    int col = ____;
    ImageView pics[]=new ImageView[row*col];

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_game);
        GridLayout g = (GridLayout)
            findViewById(R.id.grid);

        int m=0;
        for(int i=0; i<row; i++){
            for(int j=0; j<col; j++){
                pics[m]=new ImageView(this);
                setpicStart(pics[m], m);
                pics[m].setId(m);
                g.addView(pics[m]);
                m++;
            }
        }
    }
}
```

```
public void setpicStart(ImageView i, int pos){
    int x = pos/col;
    int y = pos%col;
    int picnum = soap[x][y];
    if(picnum==1)

        i.setImageResource(R.drawable.______);
    else if(picnum==2)

        i.setImageResource(R.drawable.______);
    else if(picnum==3)

        i.setImageResource(R.drawable.______);
    else if(picnum==4)

        i.setImageResource(R.drawable.______);
}
```

3. This game is called Diamond Digger. Fill in the grid information with the appropriate picture code.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#000000">

    <GridLayout
        android:layout_width="wrap_content"
        android:layout_gravity="center"
        android:layout_height="match_parent"
        android:rowCount="6"
        android:columnCount="6"
        android:id="@+id/grid">
    </GridLayout>

</LinearLayout>
```

The picture information:



The grid information:



```
public class Game extends AppCompatActivity {
    int diamond[][]= {{____,____,____},
                     {____,____,____},
                     {____,____,____},
                     {____,____,____},
                     {____,____,____}};

    int row = ____;
    int col = ____;
    ImageView pics[]=new ImageView[____*____];

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_game);
        GridLayout g = (GridLayout)
            findViewById(R.id.grid);

        int m=0;
        for(int i=0; i<____; i++){
            for(int j=0; j<____; j++){
                pics[m]=new ImageView(this);
                setpicStart(pics[m], ____);
                pics[m].setId(m);
                g.addView(pics[m]);
                m++;
            }
        }
    }
}
```

```
public void setpicStart(ImageView i, int pos){
    int x = pos/____;
    int y = pos%____;
    int picnum = ____[x][y];
    if(picnum==1)

        i.setImageResource(R.drawable.____);
    else if(picnum==2)

        i.setImageResource(R.drawable.____);
    else if(picnum==3)

        i.setImageResource(R.drawable.____);
}
```

4. For each picture, you will have a number representation in an integer grid. This tracks the picture value.

(a) Why choose an int type array? (3 reasons, no sentences)

.....

(b) What is a problem with choosing an int type array?

5. Explain why a setPicMethod is useful. (2 points, sentences)

.....

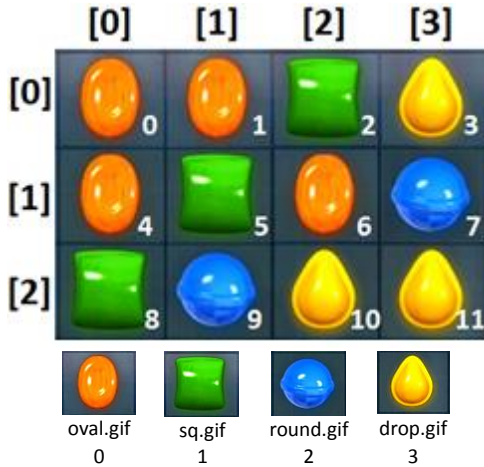
Name: _____

1D to 2D Array

0. ImageView arrays are only 1D so their IDs are unique. The 1D array for the grid below is like this:



However, if you look at the screen, it is laid out in a grid, or 2D array. We use a 2D int array to track these positions.



For the 1D ImageView array:

(a) How many elements?

.....

(b) What is the ID of the first orange oval?

.....

For the 2D int tracking array:

(c) How many rows?

.....

(d) How many columns?

.....

The references are from the 2D int tracking array. Write the corresponding ID of the 1D ImageView array.

(e) [0][0]

(f) [2][1]

(g) [3][2]

(h) [2][3]

(i) [1][3]

(j) [0][2]

(k) Using the numbers above, fill in the code for this game of candy crush.

```
public class MainActivity extends AppCompatActivity {
```

```
    int candy[][] = {{_, _, _, _}, {_, _, _, _}, {_, _, _, _}};
    int row = ____;
    int col = ____;
    ImageView pics[] = new ImageView[row * col];
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    GridLayout g = (GridLayout) findViewById(R.id.grid);
    int m = 0;
    for (int i = 0; i < row; i++) {
        for (int j = 0; j < col; j++) {
            pics[m] = new ImageView(this);
            setpic(pics[m], m);
            pics[m].setId(m);
            g.addView(pics[m]);
            m++;
        }
    }
}
```

```
public void setpic(ImageView i, int pos) {
    int x = pos / col;
    int y = pos % col;
    int picnum = ____[x][y];
    if (picnum == ____)
        i.setImageResource(R.drawable.____);
    else if (picnum == ____)
        i.setImageResource(R.drawable.____);
    else if (picnum == ____)
        i.setImageResource(R.drawable.____);
    else
        i.setImageResource(R.drawable.____);
}
```

If n is the ID of the ImageView:
 $\text{int } x = n / \text{col};$
 $\text{int } y = n \% \text{col};$

1. Assume that you have an grid that is 6 (rows) x 5 (cols).

(a) How many ImageViews do you need?

(b) Given the ImageView IDs, determine each button's (x, y), aka (row, col), position in the int tracking array.

6	10	17	29
---	----	----	----

Grid Array Algorithms

2. The following pictures are placed in arrays to produce two levels of a game as shown below.

(a) Fill in the lvl1 and sol1 array shown in the array below.



```
int row = 3;
int col = 6;
int game[][]= new int[row][col];
```

Level 1



Level 2



```
int lvl1[][]= {{_, _, _, _, _, _},
               {_, _, _, _, _, _},
               {_, _, _, _, _, _}};
```

```
int lvl2[][]= {{_, _, _, _, _, _},
               {_, _, _, _, _, _},
               {_, _, _, _, _, _}};
```

(b) This method redraws the screen with whatever is currently in the game array.

```
public void redraw() {
    int m = 0;
    for (int i = 0; i < _____; i++) {
        for (int j = 0; j < _____; j++) {
            if (_____ [i][j] == _____)
                pics[m].setImageResource(R.drawable._____);
            else if (_____ [i][j] == _____)
                pics[m].setImageResource(R.drawable._____);
            else if (_____ [i][j] == _____)
                pics[m].setImageResource(R.drawable._____);
            else
                pics[m].setImageResource(R.drawable._____);
            m++;
        }
    }
}
```

(c) Create a method to copy over the array named b into the array named a.

```
public void copyOver(int a[][], int b[][]) {
    for (int i = 0; i < _____; i++) {
        for (int j = 0; j < _____; j++) {
            a[i][j] = _____[____][_____];
        }
    }
}
```

(d) Create an onClick for a button named next that copies the lvl2 array and puts it into the game array. Then it redraws the array on the screen.

```
public _____ (_____ _____) {
    copyOver(_____, _____);
    redraw();
}
```

Searching Algorithms

3.4

Name: _____

0. Do a linear search on this array for 4.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
-10	-5	4	7	9	10	16	22	28	34	37	40	55	345	900	1002

Found in position: ____

1. Do a linear search on this array for 'm'.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
a	n	w	e	q	j	k	d	f	g	h	j

Found in position: ____

2. Do a linear search on this array for "Fish".

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]
Cat	Dog	Tree	Frog	Fish	Zoo	Ape	Kit	Ham	Two	City	Red	Yam	Bob

Found in position: ____

3. Circle the most correct answer.

- T F a) Sorting is faster than searching.
- T F b) Sorting is finding an element and searching is putting in order.
- T F c) Linear Search has one for loop so it is an "n" speed algorithm.
- T F d) Linear Search is slower than Selection Sort.
- T F e) Linear Search is faster than Quick Sort.
- T F f) Linear Search would work on unsorted data.
- T F g) Linear Search would work well when looking for the name "ZZtop" in the phone book.
- T F h) Linear Search would work well when looking through a pile of unsorted test papers to find a particular student's work.
- T F i) The Google Search engine uses linear search because it is really efficient.



4. For each of these algorithms:

- ☐ (a) **Change the code** from an integer search to a String search.
- ☐ (b) Fill in the algorithm name & speed.

<pre>int find = Integer.parseInt(in.getText().toString()); int high = array.length; int low = 0; boolean foundit = false; int mid = 0; while (high >= low && !foundit) { mid = (high + low) / 2; if (array[mid] == find) foundit = true; else if (find > array[mid]) low = mid + 1; else //if (find < array[mid]) high = mid - 1; } if(foundit) result.setText("It is in position "+mid); else result.setText("It is not in the array");</pre>	<pre>int find = Integer.parseInt(in.getText().toString()); int pos = -1; for (int i = 0 ; i < array.length ; i++) { if (array[i] == find) pos = i; } if(pos!=-1) result.setText("Didn't find it"); else result.setText("It is in position "+pos);</pre>
Algorithm Name:	Algorithm Name:
Algorithm Speed:	Algorithm Speed:

5. Trace the search of this array to find '3' using binary search. Draw the search on the array too.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]
-2	3	4	6	7	8	10	12	14	34	56	78	79	80	82

Low	High	Mid

6. Trace the search of this array to find '12' using binary search. Draw the search on the array too.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]
-3	0	1	2	3	4	5	6	7	8	9	10	12	13	14	15	17

Low	High	Mid

7. Trace the search of this array to find 'W' using binary search. Draw the search on the array too.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]
A	D	E	G	I	K	L	N	P	Q	U	V	W	Z

Low	High	Mid

8. Trace the search of this array to find 'nap' using binary search. Draw the search on the array too.

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]
abe	at	be	by	do	egg	fi	go	hi	I	moo	we	zoo

Low	High	Mid

9. Circle the most correct answer.

- T F a) Binary Search is generally a great deal faster than Linear Search.
- T F b) Binary Search relies on the data being sorted.
- T F c) A computer would search a phone book using Binary Search.
- T F d) A computer would search a library for a fiction title using Binary Search.
- T F e) A hidden cost of the fast speed of Binary Search is that the data must be sorted and sorting is a slow operation.
- T F f) Binary Search is more complex than Linear Search.
- T F g) Binary Search's speed is based on the power of 2 because the data repeatedly discards half of the data.
- T F h) If you have 100 items, in sorted order, Binary search can find an item in at most 7 tries.



10. Explain the trade-off inherent in binary search.

.....

.....

.....

Selection Sort

3.5  K

Name: _____

1. Find the maximum element in each data set.

(a) 2, 5, 67, 8, 7, 98, 1

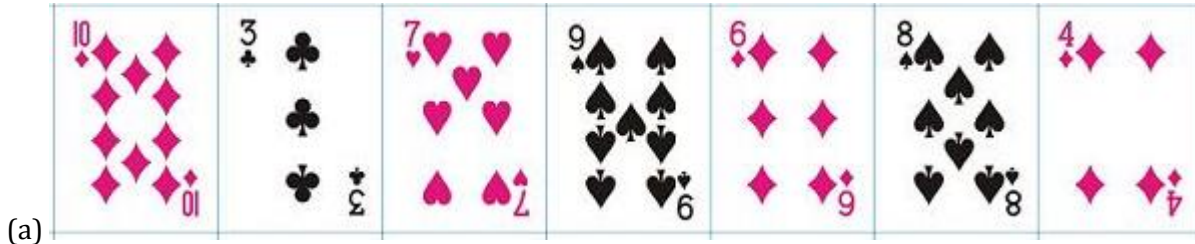
(b) a, g, r, t, y, u, p, q

(c) zebra, cheetah, elephant, hippo, rhino

(d) red, orange, yellow, green, blue, indigo, violet



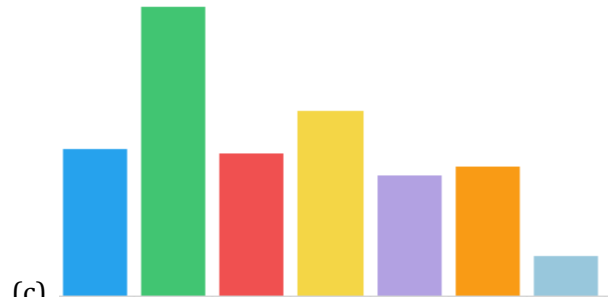
2. In each of these three data sets, circle the first two elements that swap.



(a)



(b)



(c)

3. Circle the algorithms.

Origami instructions

Paper Crane

Cake Recipe

IKEA instructions

Selection sort

Rainbow Unicorn

Cake

Turnip

Flowchart

Computer Mouse

Book

Map

4. What is the trade-off associated with Selection sort?

.....

.....

5. Put these algorithm speeds in order.

(1 is fastest, 7 is slowest)

___ $O(2^n)$

___ $O(n!)$

___ $O(\log n)$

___ Constant time

___ $O(n^2)$

___ $O(n)$

___ $O(n \log n)$

6. What speeds are these array algorithms?

(a) Linear search _____

(b) Finding array length _____

(c) Selection Sort _____

(d) Finding the average _____

(e) Mergesort _____

(f) Binary search _____

(g) Redraw _____

(h) Copy one array into another _____

7. Trace selection sort for these arrays.

(a)

8	9	4	2	3	1

(b)

V	T	Q	U	R	S

(c)

Big	Ape	Add	Cap	App

8. Look at the following code.

```
int a[] = {23, 12, 4, -4, 5, 7, 9, 55, 0, -5};

for (int left = a.length - 1 ; left > 0 ; left--) {

    int max = 0;

    for (int i = 1 ; i <= left ; i++) {
        if (a [max] < a [i])
            max = i;
    }

    int temp = a [max];
    a [max] = a [left];
    a [left] = temp;

}
```



Circle and label the following things.

- (a) code to swap two values
- (b) code to find the maximum
- (c) the loop that repeatedly finds the maximum

Identification questions:

- (a) What type is the array? _____
- (b) How long is the array? _____
- (c) What are the index numbers? ____ to ____
- (d) What is in a[2]? _____
- (e) In the outer loop, what is the loop stopping variable? _____
- (f) In the inner loop, what is the loop stopping variable? _____

9. Change the selection sort code to adapt to the following situations.

(a) The array name is num, not a. (7 changes)

```
for (int left = a.length - 1 ; left > 0 ; left--) {

    int max = 0;
    for (int i = 1 ; i <= left ; i++) {
        if (a [max] < a [i])
            max = i;
    }

    int temp = a [max];
    a [max] = a [left];
    a [left] = temp;

}
```

(b) The array type is double (only 1 change)

```
for (int left = a.length - 1 ; left > 0 ; left--) {

    int max = 0;
    for (int i = 1 ; i <= left ; i++) {
        if (a [max] < a [i])
            max = i;
    }

    int temp = a [max];
    a [max] = a [left];
    a [left] = temp;

}
```

(c) The array type is String (2 changes)

```
for (int left = a.length - 1 ; left > 0 ; left--) {

    int max = 0;
    for (int i = 1 ; i <= left ; i++) {
        if (a [max] < a [i])
            max = i;
    }

    int temp = a [max];
    a [max] = a [left];
    a [left] = temp;

}
```

(d) You want to go from largest to smallest (only 1 code change is needed, then 5 to rename max)

```
for (int left = a.length - 1 ; left > 0 ; left--) {

    int max = 0;
    for (int i = 1 ; i <= left ; i++) {
        if (a [max] < a [i])
            max = i;
    }

    int temp = a [max];
    a [max] = a [left];
    a [left] = temp;

}
```

3.6 K

0. Which tracing shows bubble sort and which show selection sort?

9	8	7	6	5
8	9	7	6	5
8	7	9	6	5
8	7	6	9	5
8	7	6	5	9
7	8	6	5	9
7	6	8	5	9
7	6	5	8	9
6	7	5	8	9
6	5	7	8	9
5	6	7	8	9

9	8	7	6	5
5	8	7	6	9
5	6	7	8	9

q	a	m	b	c
c	a	m	b	q
c	a	b	m	q
b	a	c	m	q
a	b	c	m	q

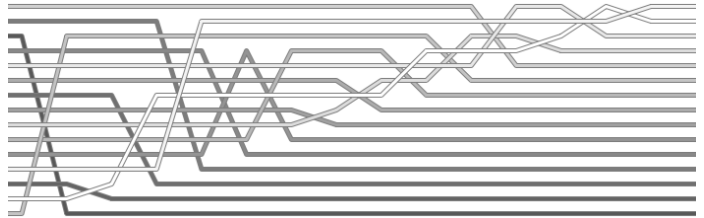
q	a	m	b	c
a	q	m	b	c
a	m	q	b	c
a	m	b	q	c
a	m	b	c	q
a	b	m	c	q
a	b	c	m	q

A row of ten black silhouettes of people of different heights standing on a horizontal line. From left to right, the heights vary, with the tallest person on the far left and the shortest person on the far right. The silhouettes represent a range of human sizes, from a small child to a tall adult.

Selection sort



Bubble sort



--

--

Bubble Sort

[illegible]

I	U	E	A	O

8	6	4	1	3

[illegible]

```
int a [] = {5, 62, 81, 9, 30, 42, 0};
```

```
public void Bubble(View view) {
    clear();
    unsort();
    bubbleSort(a);
    redraw();
}

public void bubbleSort(int a[]) {
    int n = a.length;
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - 1 - i; j++) {
            if (a[j + 1] < a[j]) {
                int temp = a[j];
                a[j] = a[j + 1];
                a[j + 1] = temp;
                addNewRow();
            }
        }
    }
}
```



4. Answer the following questions about the adjacent code.

(a) What is in the array to start?

0	1	2	3	4	5	6

(b) What type is the array?

(c) What is the name of the array?

(d) What is a.length's value?

(e) What is the onClick of the button?

(f) What is in memory like after the array is sorted?

0	1	2	3	4	5	6

(g) What are the three lines to swap the elements? Box them.

5. What is the trade-off inherent in bubblesort?

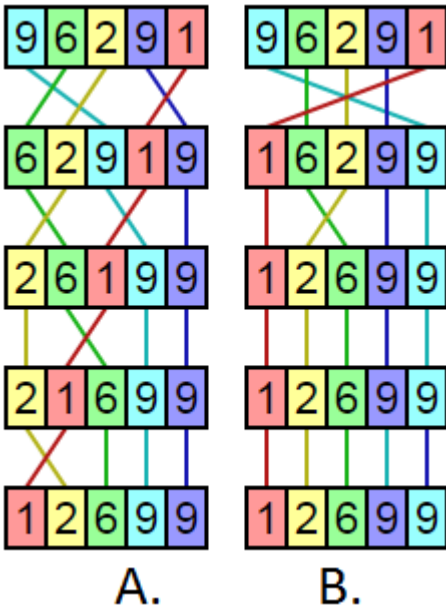
.....

.....

.....

.....

6. Which one is selection sort? (circle)



7. In 2013, the Waterloo Beaver Computing Challenge asked the following question. You are arranging people in order based on the numbers on their shirts. The order to start is:

7 3 2 9 8 5 1 4 6

You will arrange individuals using the following technique:

- Look at two consecutive people at a time, starting from the left.
- If the person on the left has a number which is larger than that of the person on the right, switch the positions of those two people; otherwise, leave them in the order they are in.
- Move to the right one position, so that you are comparing one new person with one of the people just compared, and repeat the above comparison and potential swap.

Once you have compared the right-most two people in the list, we call this one "pass" over the list.

How many passes over the list are required until the list is in the order:

1 2 3 4 5 6 7 8 9?

Circle the best answer: (a) 2 (b) 4 (c) 6 (d) 9

Merge Sort

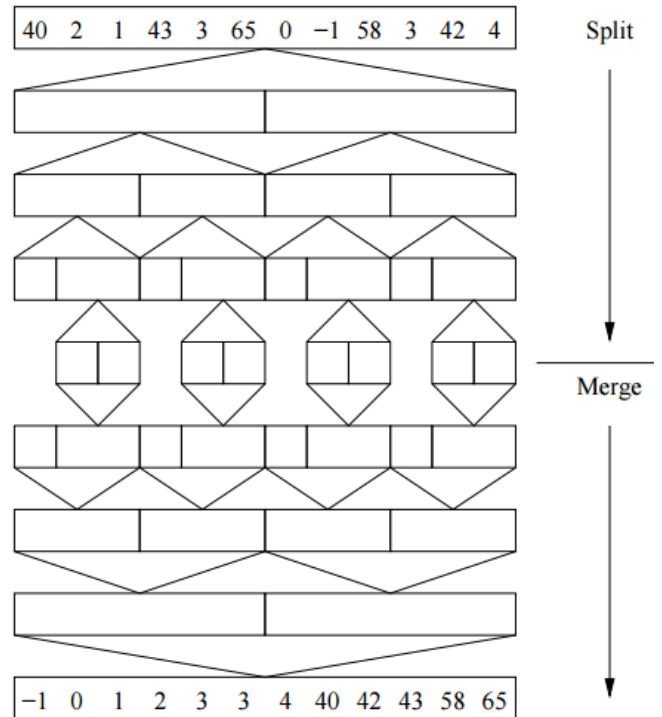
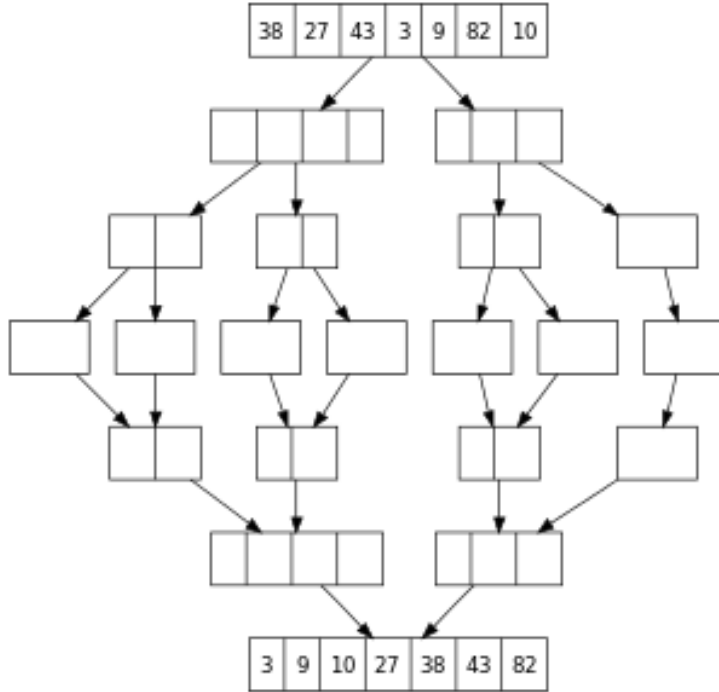
3.7 

Name: _____

Conceptually, a merge sort works as follows:

- Divide the unsorted list into n sublists, each containing 1 element (a list of 1 element is considered sorted).
- Repeatedly merge sublists to produce new sorted sublists until there is only 1 sublist remaining. This will be the sorted list.

0. Fill in both tracings of mergesort:



1. Fill in the speeds of the algorithms:

- _____ Bubble sort, best case.
- _____ Bubble sort, average case.
- _____ Delete to a list.
- _____ Add to a list.
- _____ Mergesort average case.
- _____ Selection sort, average case.
- _____ Print out array.
- _____ Find the maximum

2. Given the following array of numbers:

{21, 1, 26, 45, 29, 28, 2, 9, 16, 49, 39, 27, 43, 34, 46, 40}
which answer illustrates the array to be sorted after 3 recursive calls to mergesort? (circle the correct answer)

- (A) {16, 49, 39, 27, 43, 34, 46, 40}
- (B) {21, 1}
- (C) {21, 1, 26, 45}
- (D) {21}

3. Consider pseudocode of the main method of mergesort.

```
public int[] mergeSort(int[] array) {
    if (array.length <= 1)
        return array;
    else {
        int middle = array.length / 2;
        int firstHalf = mergeSort(array[0..middle - 1]);
        int secondHalf = mergeSort(array[middle..array.length - 1]);
        return merge(firstHalf, secondHalf);
    }
}
```

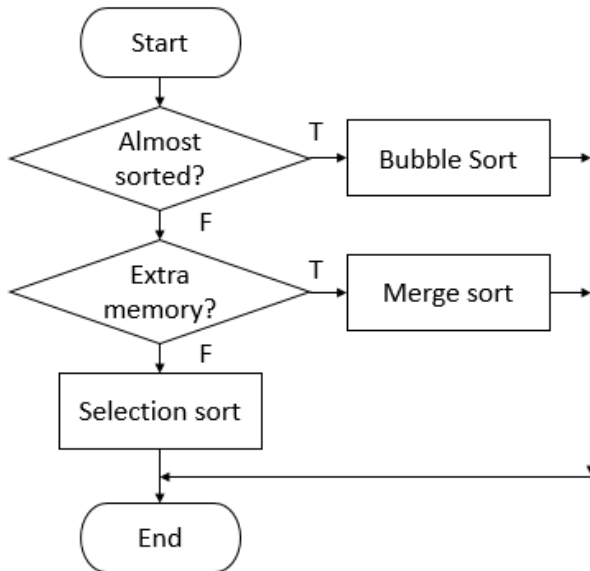
- (a) Circle and label the base case.
- (b) Box the recursive cases.
- (c) What is the name of the method?
- (d) Another method is called (outside of mergeSort). What is it?
- (e) What is the return type?
- (f) What is the parameter type?

4. Answer true or false about each statement.

- T F a) Mergesort is recursive; that is why it is fast.
- T F b) The division and merging process in mergesort requires extra memory.
- T F c) The base case of mergesort is an array of one element.
- T F d) Once the base case is reached in mergesort, the division process begins.
- T F e) Once the base case is reached in mergesort, the merge process ends.
- T F f) "Merge" means a smooth, orderly joining of two things.
- T F g) Mergesort is an in-place algorithm.
- T F h) Selectionsort is an in-place algorithm.
- T F i) Mergesort is faster than Quicksort.
- T F j) Mergesort is faster than Bubble sort when the array is almost sorted.

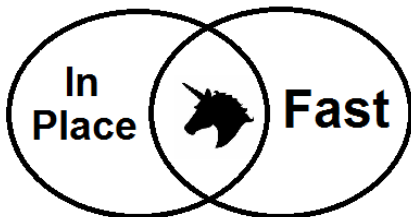


5. Which sorting algorithm should be used in each instance?



- (a) You have a random array, but no extra memory.
.....
- (b) You have an almost sorted array, but no extra memory.
.....
- (c) The array is sorted, but one item was added to the front. You have no extra memory.
.....
- (d) You have 500 million elements in random order.
.....
- (e) You have an array of 500 names in random order.
.....
- (f) You have an array of 500 names that is almost in correct order.
.....

6. Explain the trade-off of mergesort.



.....

.....

.....

.....

.....

7. Sort each array using mergesort. (Make boxes and use lines to connect them)

6 3 2 8 4 9 5 0

9 1 6 4 7 3 8 2

Quick Sort

3.8 

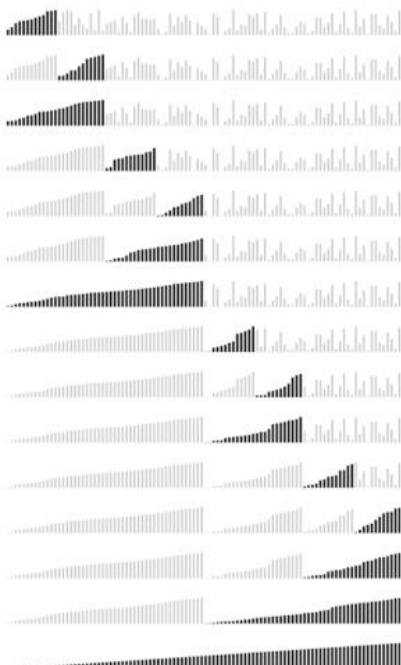
Name: _____

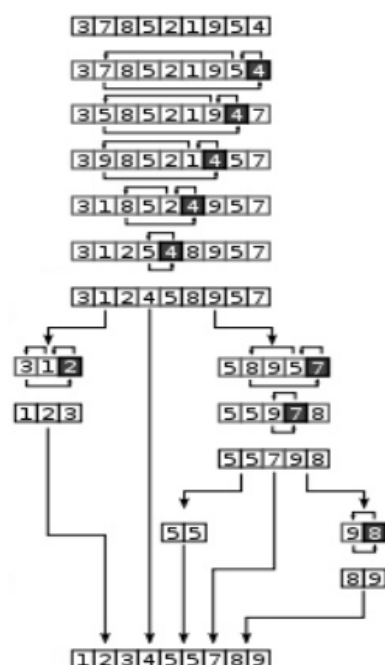
```
void quicksort(int[] array, int startIndex, int endIndex)
{
    if (startIndex >= endIndex) {
        return;
    }
    else {
        int pivotIndex = partition(array, startIndex, endIndex);
        quicksort(array, startIndex, pivotIndex - 1);
        quicksort(array, pivotIndex + 1, endIndex);
    }
}
```

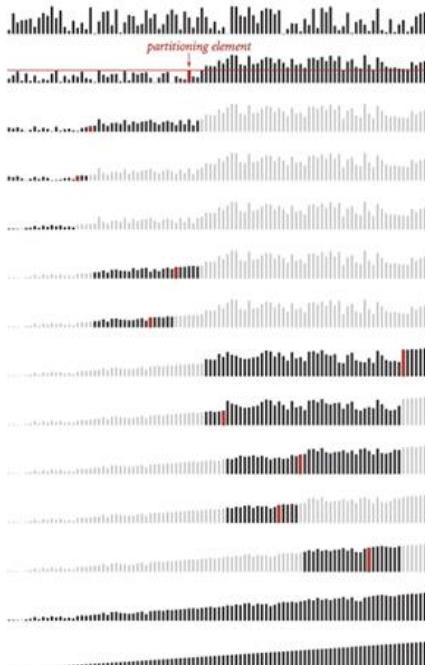
The steps are:

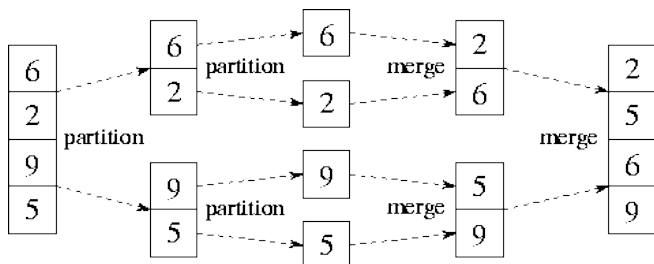
- Pick an element, called a pivot, from the array.
- Partitioning: reorder the array so that all elements less than the pivot come before it, and all elements greater come after. After partitioning, the pivot is in its final position.
- Recursively apply the above steps to the sub-array of elements before the pivot AND again to those after it.

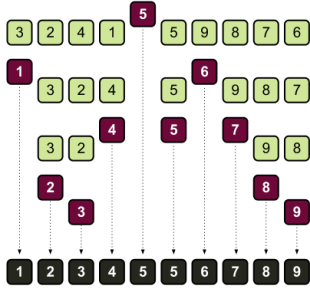
0. Which of the following pictures show quickSort and which show mergeSort?

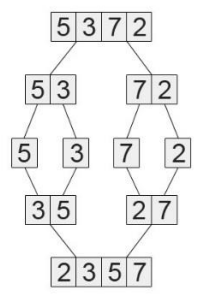












1. Circle True or False for each question.



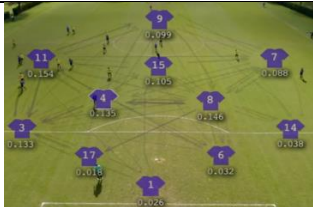



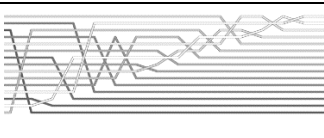


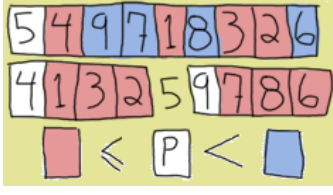
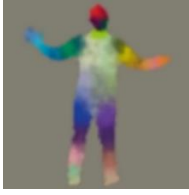

- T F a) The O in $O(n)$ stands for order.
- T F b) The n in $O(n)$ stands for number of operations.
- T F c) The n in $O(n)$ stands for number of elements in the array.
- T F d) The n in $O(n)$ stands for the number of seconds to run the algorithm.
- T F e) Algorithm speed isn't measured in terms of time, because that is hardware dependant.
- T F f) Constant time is another way of saying $O(1)$.
- T F g) The fastest time is $O(\log n)$.
- T F h) Quicksort is a faster $O(n \log n)$ than Mergesort, assuming a randomized data set.

Name: _____

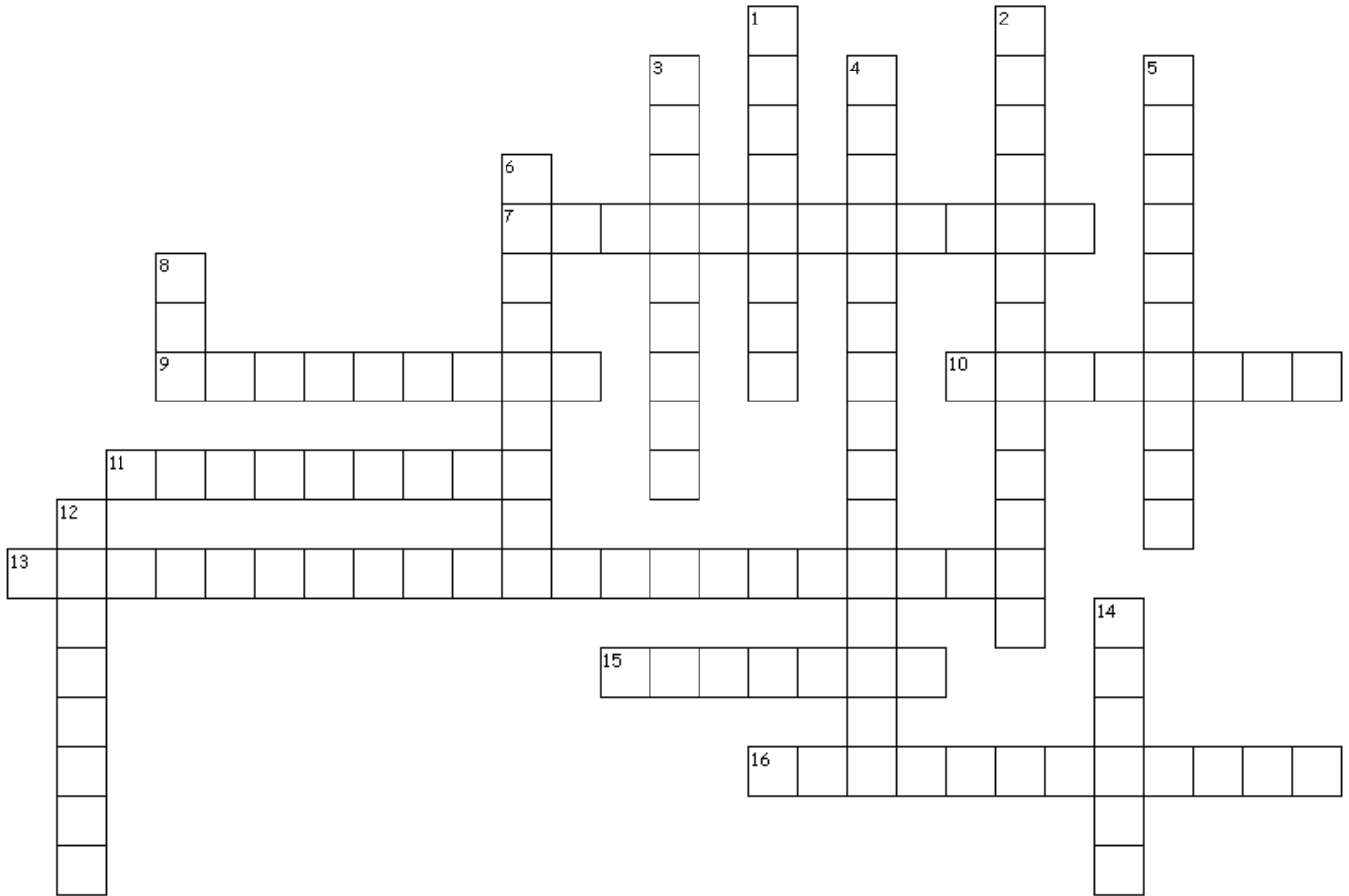
1. From the movie, the Secret Lives of Algorithms, fill in the first column. (The last 5 are from class)

Word	Description
	A series of steps to complete a task.
	Phones use this to quickly find human faces in real time on the screen.
	A game strategy algorithm. If your opponent takes n items, you take $(4-n)$.
	The oldest recorded algorithm. Finds the greatest common divisor for 2 #s.
	A sorting algorithm that repeatedly swaps out of order elements into place.
	John Von Neumann's sorting algorithm. Data is divided then merged.
	A billion dollar algorithm that puts searches in an order useful to the user.
	A gaming system that taught itself how to understand human movement.
	A streaming service that uses algorithms to make future recommendations.
	Robots that use algorithms to put together customer orders for mailing.
	A compromise. You give up something to get something else.
	Looks at each item in the list until the desired item is found.
	The worst sorting algorithm. It might never finish.
	Repeatedly discarding half of a sorted list until the item is found.
	A sorting method where you repeatedly swap the max into place.
	The fastest in-place algorithm in the general case.

2. Write the algorithm name that matches each picture.

3. Fill in the crossword using terms from this unit.

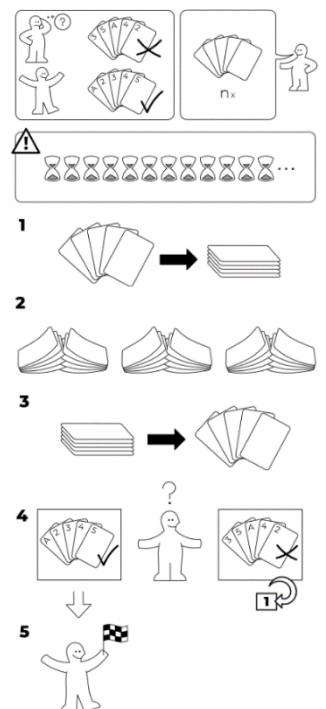


Across

7. A searching algorithm that always works, but it is a bit slower.
9. A sorting algorithm invented by John Von Neumann. Data is divided and merged in order.
10. The worst sorting algorithm.
11. The fastest in-place sorting algorithm for regular randomized data.
13. The oldest recorded algorithm. Finds the greatest common divisor for 2 large numbers.
15. A streaming service that uses algorithms to make recommendations based on your past movie choices.
16. The fastest searching algorithm.

Down

1. Google's billion-dollar algorithm; puts web searches in an order useful to the user.
2. A sorting method where you repeatedly find the max and swap it into place.
3. Robots that use algorithms to efficiently put together customer orders for mailing.
4. Phones use this algorithm to quickly find human faces in real time on the screen.
5. A sorting algorithm that repeatedly swaps out of order elements into place. Named after pop bubbles.
6. A series of steps to complete a task.
8. A game strategy algorithm. If your opponent takes n items, you take $(4-n)$ items.
12. A compromise made when you make a choice. You give up something to get something else.
14. A gaming system that taught itself how to understand human movement.



3.10

Name: _____

1. What does the acronym PDLC stand for?

--	--	--	--

2. Match the job with the phase of the PDLC.

*Ad Writer, Alpha Tester, Analyst, Beta Tester, Designer, Field Service Technician, Graphic Artist, Help Desk
Lead Programmer, Musician, Programmer, Sales Analyst, Support and Training, Writer*

Software Development Lifecycle Phases			
Analysis	Design	Code	Reflection/Maintenance

3. Unscramble these PDLC jobs.

LADE GERMOARRMP

SUORTPP NAD NRTIANGI

CPGAHRI TARSIT

Two empty ten-frames, each consisting of a 2x5 grid of squares, are provided for base ten blocks.

ELFİD CEVRESİ HANCİTCİNE

REITWR

		22		7	

SAESL SYNLTATA

AD REIRWT

18

HAPI_A RESET

--	--	--	--	--

--	--	--	--	--	--

CAMNIISU

13	4						25

REESDNIG

A horizontal number line with 14 equal segments. The number 14 is written below the first segment.

NATLAYS

			10	23		

TAFB TITERES

--	--	--	--

--	--	--	--	--	--

MAROGMEPRR

		11				8		21	

LEHP KEDS

1	2	3	4	5	6

7	8	9	10	11	12	13	14	15	16

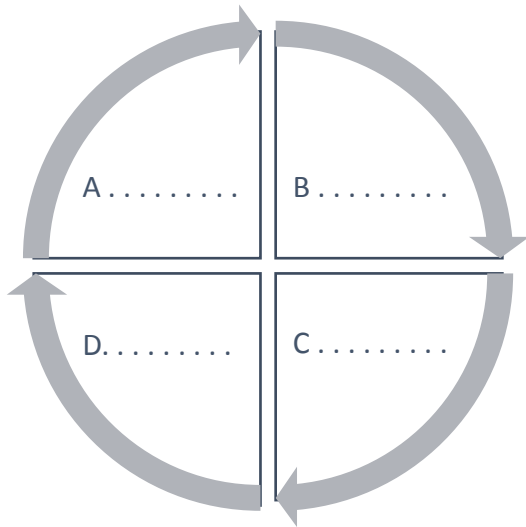
1	17	18

19	20	21	22	23	24	25	26

4. Fill in the job that matches the description.

	(a) Answers questions that people email or call in about the software.
	(b) Helps people use their new software on site.
	(c) Decides the overall game or software idea.
	(d) Draws the flow charts, screen flow diagrams and structure charts.
	(e) Someone in the company, with some programming knowledge who does testing.
	(f) A user who does testing.

5. What are the 4 PDLC phases?



6. Cross out the items that do not belong in the phase.

Analysis

1. Create a storyline
2. Hold Focus Groups
3. Design levels
4. Pitch Concept
5. Write while loops
6. Eat Lunch
7. Define the Problem
8. Build a snowman
9. Analyze sales figures
10. Write a list of specifications

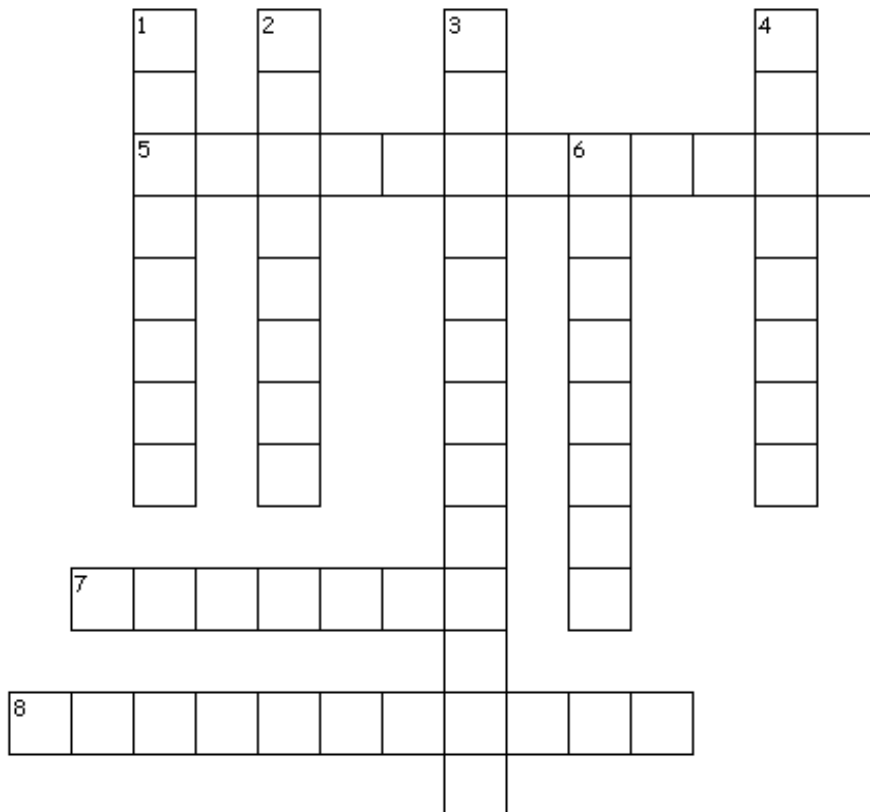
Design

1. Character Design
2. Artwork
3. Level Design
4. Watch Netflix
5. Make Prototypes
6. Create storyline
7. Alpha testing
8. Draw Flow charts, structure charts
9. Write and record music
10. Define the problem

7. True or False: circle the most correct answer.

- T F a) Writing loops comes before Flowcharts in your code.
- T F b) Testing takes $\frac{3}{4}$ of the coding phase in a professional game.
- T F c) The PDLC is necessary because big programs require a lot of planning to enable large teams to work together.
- T F d) An analyst is generally better paid than a programmer, because the analyst job is harder.
- T F e) A tester is generally better paid than a programmer, because it is difficult to find coding errors.

8. Fill in the crossword using job titles from the PDLC.



Across

5. Reviews the sales in the reflection phase. Thinks about a sequel.
7. Would pitch the overall idea and decide the direction of the software.
8. A type of testing done inside the company.

Down

1. Draws diagrams and lays out the details of the program
2. Supports users who are having difficulties with the software.
3. Designs the screen layout, colours, character design and backgrounds.
4. Composes and records music for the software.
6. creates advertisements for the software in the reflection phase