

Unit 5 ~ Arrays & Loops

Unit 5A

Arrays are a great way to store lots of data of the same type. An array is a sequence of elements, each with an index number associated with it.

- 📖 The first element in an array is at index 0, the second at index 1, the third at index 2, and so on.
- 📖 The last index of the array will always be length of the array minus 1.

The reason that the index starts at 0 is because we are actually measuring the distance to the start of the array. It is like you are in a lineup...a queue. If you are at the front, you are the next person in...you are at position 0. There is NO ONE in front of you.

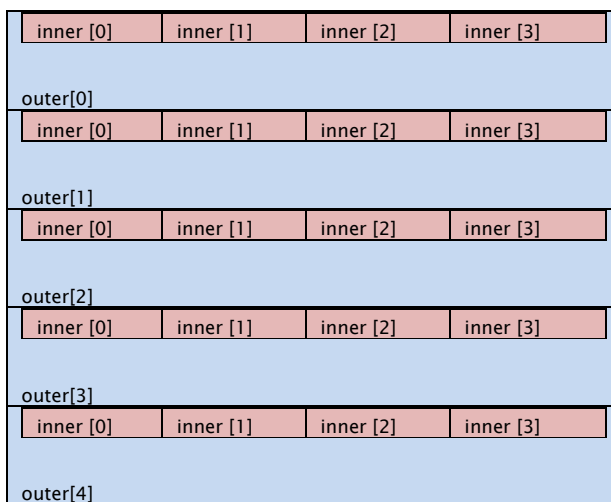
To declare an array in Java, you may follow these 2 examples:

- `double [] grades;`
- `double grades [];`

Either of the above works.

2D Arrays A 2D array is also known as a **matrix**. A matrix is a collection of elements arranged like a table...with a row and a column. Think about the game Battleship. This is an example of a 2D array.

Java creates multidimensional arrays as arrays of arrays. The first index is the location of the outer array. The second index is the location in the inner array.



Refer to elements in a matrix with this format:

- `arr [outer index] [inner index]`

Iteration is a structure that repeats a block of code a certain number of times or until a certain condition is met.

The while Loop

In a while loop you test a condition and, as long as that condition is true, the program performs an action.

```
while (condition ) {  
    statements;  
}
```

The following while statement executes _____ times.

```
int num = 0;  
while (num <5) {  
    num +=1;  
}
```

The do-while Loop

This alternative to the while loop does not evaluate the condition until the loop runs at least once.

```
do {  
    statements;  
} while (condition);
```

The following loop will continue to prompt the user until a number between 1 and 10 is entered:

```
do {  
    System.out.print("Enter a number between 1 and 10 inclusive: "  
);  
    number = input.nextInt();  
} while (number <1 || number >10);
```

Infinite Loops

If poor logic is used to code, then it is possible to create an infinite loop...one in which the exiting condition is never met.

<pre>int num = -1; while (num <0) num = -1;</pre> <p>//infinite loop</p>	<pre>while (num <0); num +=1;</pre> <p>//infinite loop</p>	<pre>int num = 1; do { num +=1; } while (num >=0);</pre> <p>//overflow error</p>
---	---	---

Eventually the application will *hang* or get an *overflow error*.

5A Worksheet 1¹

1. What is the output?

```
int x=1;
while(x<5)
{
    x++;
    System.out.print(x);
}
```

2. What is the output?

```
int m=2, total=0;
while(m<6)
{
    total=total+m;
    m++;
}
```

3. What is the output?

```
int z=2, sum=0;
while(z<9)
{
    z++;
    sum=sum+z;
}
System.out.print(sum);
```

4. What is the output?

```
int k=3;
String s="";
while(k>-1){
    s=k+" "+s;
    k--;
}
System.out.print(s);
```

5. What is the output?

```
int b=5;
String list="";
while(b<11)
{
    b=b+2;
    if(b%2==1)
        list=b+" "+list;
}
System.out.print(list);
```

¹ whileloopsworksheet1

Your turn

View: Prompter

Create a Prompter application that prompts the user for two numbers. The first number is a min value and the second is a max value. Prompter then prompts the user for a number between the min and max numbers entered. The user should be continually prompted until a number within the range is entered. Be sure to include the min and max numbers in the prompt.

Accumulators

Many algorithms require counting and summing of values.

```
Prompt user for a value
while (value != 0)
    count value
    add value to sum of values
    prompt user for another value
Display average of values (sum/count)
```

A program that counts the number of values entered by the user is actually counting the number of loop iterations.

```
numValues +=1;
```

An assignment statement to sum values entered could be:

```
sumValues += newValue;
```

or

```
sumValues = sumValues + newValue;
```

Another example:

```
rolls = 0;
counter = 0;
while (counter < 10)
{
    // Roll a simulated die
    roll += 1;
    if (myRandom.nextInt(6) + 1 == 6)
    {
        counter += 1;
    }
}
```

This loop repeats while the counter variable is less than 10. The counter variable is incremented each time a simulated die rolls a 6. The roll variable tells you how many rolls of the die were needed to reach 10 sixes.

Another example:

```
sum = 0;
counter = 0;
do
{
    // Roll a simulated die
    sum += myRandom.nextInt(6) + 1;
    counter += 1;
}
while (sum <= 30);
```

This loop rolls a simulated die **while** the **sum** of the rolls does not exceed 30. It also keeps track of the number of rolls (**counter**) needed to achieve this **sum**.

Flag Sentinel

Often a constant declared as SENTINEL stores a *flag* or *sentinel* to signify that the loop should stop iterating.

```
while (value != SENTINEL) {
    statements;
}
```

Your Turn

Review: Evens

Create an Evens application that displays the even numbers between 1 and 20, inclusive.

Your Turn

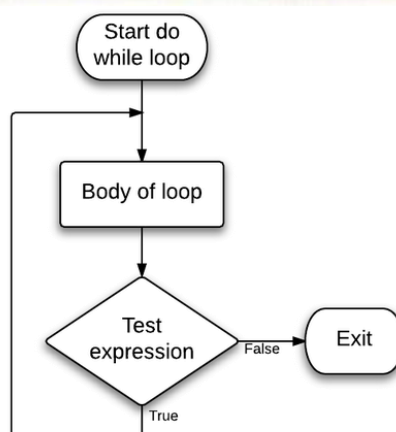
Review: NumbersSum

Create a NumbersSum application that prompts the user for a number and then displays the numbers 1 through the number entered, each on a separate line. Below the numbers, the sum is displayed.

Your Turn

Review: PercentPassing

Create a PercentPassing application that prompts the user for a set of scores and then calculates the percentage of scores above 70%. The user should have the option to enter as many scores as needed. (Hint: Use an `if` statement and another counter.)



Lab 09A – LOOP CIRCLES 2

Lab Goal : This lab was designed to teach you how to use a while loop.

Lab Description : Complete the `drawCircles()` method so that it will draw 9 or more circles as in the image below. The start up program only draws the first and smallest circle.

Basic while loop syntax :

```
int x = 200;
while(x>90)
{
    //do something
    x = x-10;
}
```

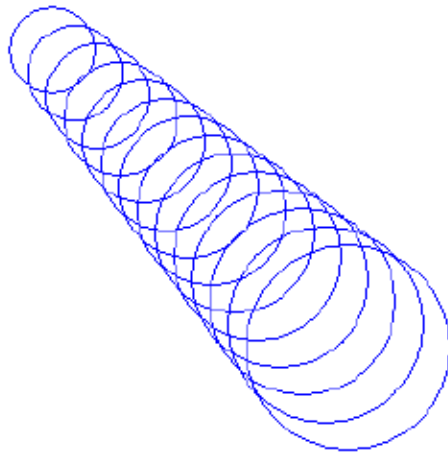
Files Needed ::

`GraphicsRunner.java`
`WhileLoopCircles.java`

Sample Output :

Lab 9A

Drawing Circles Using a while loop



5A Worksheet 2²

1. What is the output?

```
int x=1;
do{
    x++;
    System.out.print(x);
}while(x<5);
```

2. What is the output?

```
int m=2, total=0;
do{
    total=total+m;
    m++;
}while(m<6);
System.out.print(total);
```

3. What is the output?

```
int z=2, sum=0;
do{
    z++;
    sum=sum+z;
}while(z<9);
System.out.print(sum);
```

4. What is the output?

```
int k=3, tot=0;
do{
    tot=tot+k;
    k++;
}while(k<11);
System.out.print(tot);
```

5. What is the output?

```
int b=5, ans=0;
do
{
    b=b+2;;
    ans=ans+b;
}while(b<11);
System.out.print(ans);
```

² dowhileloopsworksheet

Lab Goal : This lab was designed to teach you how to use a do while loop.

Lab Description : Complete the `drawCircles()` method so that it will draw 9 or more circles as in the image below. The start up program only draws the first and smallest circle.

Basic do while loop syntax :

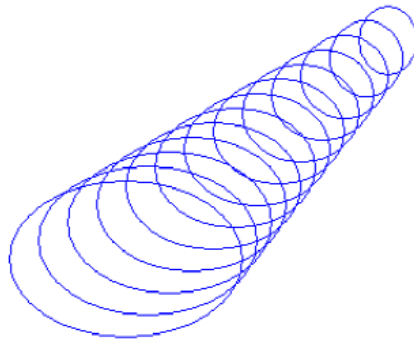
```
int x = 200;  
do{  
    //do something  
    x = x-10;  
}while(x>90);
```

Files Needed ::
`GraphicsRunner.java`
`DoWhileLoopCircles.java`

Sample Output :

Lab 9B

Drawing Circles Using a do while loop



Lab Guessing Game 1

Write an app that has the user repeatedly guess for the randomly chosen secret number between 20 and 50. Keep track of the number of guesses.

Lab PrimeNumber

- A prime number is an integer greater than 1 that is evenly divisible by only 1 and itself. For example, 2, 3, 5, and 7 are prime numbers, but 4, 6, 8, and 9 are not. Create a PrimeNumber application that prompts the user for a number and then displays a message indicating whether the number is prime or not. Hint: The `%` operator can be used to determine if one number is evenly divisible by another.
- Modify the application to prompt the user for two numbers and then display the prime numbers between those numbers.

Lab Reverse Number³

Lab Goal : This lab was designed to teach you how to use a while loop.

Lab Description : Write a program that will take an integer and reverse it. You must use a while loop and % to perform the reverse.

Sample Data :

234
10000
111
9005
84645
8547
123456789

Files Needed ::

ReverseNumber.java
Lab09c.java

Sample Output :

234 reversed is 432
10000 reversed is 1
111 reversed is 111
9005 reversed is 5009
84645 reversed is 54648
8547 reversed is 7458
123456789 reversed is 987654321

algorithm help

```
local variable rev = 0
local variable num = original number
while(num > 0)
{
    rev = rev times 10 + num % 10
    num = num divided by 10
}
```

Critical Thinking Questions

1. Explain the difference between a while and a do-while loop.

2. How many times will this loop execute?

```
int x = 0;
do {
    x = x + 2;
} while (x < 120);
```

3. What initial value x would make this loop infinite?

```
do {
    x = x - 3;
} while (x < 120);
```

³ Lab 09c

Boolean Worksheet 1

1. What is the output?

```
boolean x = true;
boolean y = false;
out.println(x&&y);
out.println(x | y);
out.println(!(x&&y));
out.println(!(x | y));
```

2. What is the output?

```
int a = 3;
int b = 7;
if(a>5 && b>5)
    out.println("both are >5");
out.println("testing");
```

3. What is the output?

```
int c = 9;
int d = 10;
if(c>5 && d>5)
    out.println("both are >5");
out.println("testing");
```

4. What is the output?

```
int e=8;
int f=6;
do{
    e--;
    f--;
}while(e>5 && f>5);
out.println(e + " - " + f);
```

5. What is the output?

```
int g = 14;
int h = 12;
do{
    g--;
    h--;
}while(g>5 || h>5);
out.println(g + " - " + h);
```

while (! (succeed == try()));



Lab Divisors⁴

Lab Goal : This lab was designed to teach you how to use while and do while loops.

Lab Description : Generate all of the divisors for any given number. 6 has the divisors 1, 2, and 3. Use mod (%) to find the divisors.

Sample Data :

10
45
14
1024
1245
33
65535

Files Needed ::

Divisors.java

Lab09e.java

Sample Output :

10 has divisors 1 2 5
45 has divisors 1 3 5 9 15
14 has divisors 1 2 7
1024 has divisors 1 2 4 8 16 32 64 128 256 512
1245 has divisors 1 3 5 15 83 249 415
33 has divisors 1 3 11
65535 has divisors 1 3 5 15 17 51 85 255 257 771 1285 3855 4369 13107 21845

Copy and paste this lab for marks to Klimack Marking

Tracing a Program

Variable traces are an effective manual debugging tool. They list the values of variables at points of assignment.

Trace the code and show the output

b	sum	output
2	0	

```
int b=2, sum=0;
while(b<11)
{
    b++;
    if(b%2==1)
        sum=sum+b;
}
System.out.print(sum);
```

Unit 5 Quiz 1

While and do-while loops




⁴ Lab 09e

Unit 5B

If you know beforehand how many times the loop is supposed to iterate, then use a **For Loop**.

Format:

```
for (initialization; condition; increment) {  
    Statements;  
}
```

-  The initialization occurs ONCE.
-  The condition is checked. If it is true, the statements are executed.
-  The increment occurs.

Example:

```
for (int x = 1; x <= 10; x++) {  
    System.out.println("x : " + x);  
}
```

Note: The *scope* of control variable x in this example is the lifetime of the loop. When the final } is reached, x is garbage.

Expressions Using ++ or --

If the ++ or -- operator appears before the operand, it is called *prefix* (i.e. ++i). An operator after the operand is called *postfix* (i.e. i++). Either operator location has the same effect on the final value of the operand. However, in an expression, the prefix version uses the value of the operand *after* the operation. For example, when x is 12, the statement ++x returns 13. In the postfix version, x++, 12 is returned. Therefore, a statement such as x++ >= 13 is false and could be ambiguous. Using the ++ and -- operators in an expression is poor programming style.

The loop may also count down, or increment/decrement by any integer value.

Note: Although you may modify the control variable from within the loop, this is poor programming style.

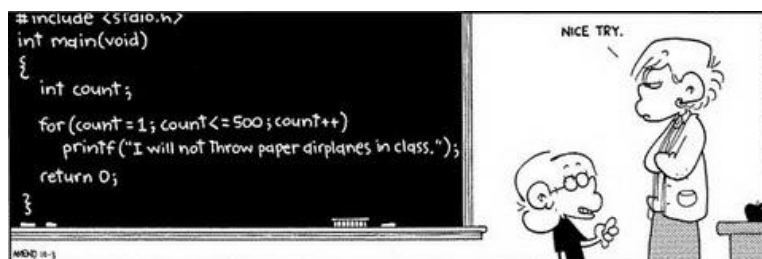
YouTube

Review: Factorial

Create a Factorial application that prompts the user for a number and then displays its factorial. The factorial of a number is the product of all the positive integers from 1 to the number. For example, $5! = 5 \times 4 \times 3 \times 2 \times 1$.

Review: OddSum

Create an OddSum application that prompts the user for a number and then sums the odd numbers from 1 to the number entered.





Review: Variable Trace

Using paper and pencil, create a variable trace for the following code, tracing the values of num1, num2, i, and any output:

```
int num1 = 0;
int num2 = 0;
for (int i = 0; i <= 4; i++) {
    num1 = i * i;
    num2 += num1;
    System.out.print(num1 + " ");
}
System.out.println(num2);
```

Break and Continue

You may exit a **for loop** early using a break statement. This will transfer program control to the statement following the closing brace. Use of a continue statement will skip all statements remaining in the loop and return program control to the **for** statement.

5B Worksheet 1⁵

1. What is the output?

```
for(int i=1; i<15; i=i+3)
{
    out.println(i);
}
```

2. What is the output?

```
for(int j=11; j>-2; j=j-2)
{
    out.println(j);
}
```

3. What is the output?

```
for(int x=20; x<40; x=x+3)
{
    out.println(x);
}
```

4. What is the output?

```
for(int m=30; m>0; m=m-4)
{
    out.println(m);
}
```

5. What is the output?

```
int total=0;
for(int s=1; s<15; s++)
{
    total=total+s;
}
out.println(total);
```

⁵ forloopworksheet

Lab 08A – LOOP CIRCLES 1

Lab Goal : This lab was designed to teach you how to use a for loop.

Lab Description : Complete the `drawCircles()` method so that it will draw 9 or more circles as in the image below. The start up program only draws the first and smallest circle.

Basic for loop syntax :

```
for(int x=90; x<200; x=x+20)
{
    //do something
}
```

Files Needed ::
`GraphicsRunner.java`
`ForLoopCircles.java`

Sample Output :

Lab 8A

Drawing Circles Using a for loop



Lab DiceRolls⁶

Create a DiceRolls application that displays five rolls of two dice where each die is numbered from 1 to 6. The application should also show the total of each roll:

Dice 1	Dice 2	Total
6	1	7
6	4	10
3	3	6
3	5	8
3	1	4

⁶ Ch 6 LVP

Lab 08B – FOR LOOP RUNNER

Lab Goal : This lab was designed to teach you how to use a for loop.

Lab Description : Demonstrate the workings of a for loop.

```
//initialization-start           //condition-stop           //incremenation
for ( int i = 0;                  i < 10;                i = i + 1 )
{
    //do something
}
```

ANATOMY OF A FOR LOOP

Sample Data :

```
2 90 5
3 76 4
-10 8 2
5 30 2
100 150 5
```

Files Needed ::

```
ForLoopDemo.java
Lab08b.java
```

Sample Output :

```
start 2 : stop 90 : increment 5
2 7 12 17 22 27 32 37 42 47 52 57 62 67 72 77 82 87

start 3 : stop 76 : increment 4
3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63 67 71 75

start -10 : stop 8 : increment 2
-10 -8 -6 -4 -2 0 2 4 6

start 5 : stop 30 : increment 2
5 7 9 11 13 15 17 19 21 23 25 27 29

start 100 : stop 150 : increment 5
100 105 110 115 120 125 130 135 140 145
```

Lab PowersTable⁷

Create a PowersTable application that displays a table of of powers similar to:

x ¹	x ²	x ³	x ⁴	x ⁵
1	1	1	1	1
2	4	8	16	32
3	9	27	81	243
4	16	64	256	1024
5	25	125	625	3125
6	36	216	1296	7776

⁷ Ch 6 LVP

For Loop Worksheet⁸

1. What is the output?

```
for(int i=1; i<6; i=i+2)
{
    out.println(i);
}
```

2. What is the output?

```
for(int j=6; j>-2; j=j-2)
{
    out.println(j);
}
```

3. What is the output?

```
for(int k=1; k<12; k=k+3)
{
    out.println(k);
}
```

4. What is the output?

```
for(int m=11; m>0; m=m-4)
{
    out.println(m);
}
```

5. Trace the code at right.

<u>x</u>	<u>total</u>	<u>output</u>
----------	--------------	---------------

```
int total=0;
for(int x=1; x<12; x=x+2)
{
    total=total+x;
}
out.println(total);
```

Unit 5 Quiz 2

For loops

Unit 5C

⁸ Loop quiz 1a

More on Strings


The String class is large, filled with methods used to manipulate and examine Strings. Here are some common methods.

String Class	
Frequently used methods	
Name	Use
length()	returns the number of chars in string
isEmpty()	returns true if length of string is 0 else returns false
substring(int start, int end)	returns a substring of string starting at start and ending 1 char before end
substring (int start)	returns a substring starting at start and til the end of the string
toLowerCase()	returns a copy of string all lowercase
toUpperCase()	returns a copy of string all UPPERCASE
trim()	returns a copy of string with leading and trailing spaces removed
replaceFirst(String s1, String s2)	returns a string with 1 st occurrence of s1 replaced with s2
replaceAll(String s1, String s2)	returns a string with all occurrences of s1 replaced by s2
equals(String s)	returns true if the strings are the same
equalsIgnoreCase(String s)	same as equals but ignores case
compareTo(String s)	returns 0 when they are the same, a negative if s is alphabetically after the calling string , and a positive when s comes before the string
compareToIgnoreCase(String s)	same as above except ignores case
indexOf(String s)	returns the integer corresponding to the first occurrence of s in the string, else returns -1 if not found
lastIndexOf(String s)	returns the integer corresponding to the location of the last occurrence of s in the string else returns -1
startsWith(String s)	returns true when string begins with s else returns false
endsWith(String s)	returns true when string ends with s else returns false

The position of a character in a string is called its *index*. The first character is at **index 0**. The last character will then be **length()-1**.

A String is **immutable** because it cannot be changed. When you manipulate a string and change its appearance, you create a new string in memory and the reference changes from the original string to the new one.

A string is assigned **null** until it is assigned something else. Calling a method from a null String object generates the exception **NullPointerException**.



Review: FormalGreeting

Create a FormalGreeting application that prompts the user for his or her name, including title. The application should display "Hello, sir." if the string starts with Mr., "Hello, ma'am." if the string starts with Ms., Mrs., or Miss, and "Hello, name." otherwise where *name* is the user's name.

String Worksheet 2

DIRECTIONS : Fill in each blank with the correct answer/output. Assume each statement happens in order and that one statement may affect the next statement.

```
char cOne = 'o';
char cTwo = 'l';
String sOne = "hello there";
String sTwo = "hallo there";
```

```
out.print( sOne.indexOf('h') );           // LINE 1
out.print( sOne.indexOf('7') );           // LINE 2
out.print( sTwo.indexOf('a') );           // LINE 3
out.print( sTwo.indexOf(cTwo) );          // LINE 4
out.print( sOne.indexOf("lo") );          // LINE 5
out.print( sTwo.indexOf("al") );          // LINE 6
out.print( sOne.charAt(3) );              // LINE 7
out.print( sOne.charAt(0) );              // LINE 8
out.print( sOne.substring(3,6) );         // LINE 9
out.print( sOne.substring(0,4) );         // LINE 10
out.print( sOne.equals(sTwo) );           // LINE 11
out.print( sOne.compareTo(sTwo) );        // LINE 12
if first isn't 0 than stop there
out.print( sTwo.compareTo(sOne) );        // LINE 13
out.print( sTwo.compareTo("abc") );       // LINE 14
out.print( sTwo.replaceAll("e","#") );    // LINE 15
out.print( sTwo.replaceAll("#","*") );    // LINE 16
doesn't change it just sout the changed string
out.print( sTwo.length() );              // LINE 17
out.print( sOne.length() );              // LINE 18
out.print('7' - '0');                    // LINE 19
out.print('5' - 48 );                    // LINE 20
out.print(sOne.charAt(20) );             // LINE 21
```

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____
7. _____
8. _____
9. _____
10. _____
11. _____
12. _____
13. _____
14. _____
15. _____
16. _____
17. _____
18. _____
19. _____
20. _____
21. _____



Now: AccountSetup

Create an AccountSetup application that prompts the user for a user name and a password. The application should prompt the user until a password with at least eight characters is entered. The user name and password should be converted to all lowercase letters and then an appropriate message displayed. Application output should look similar to:

```
Enter a user name: WHIZKID
Enter a password that is at least 8 characters: Guess
Enter a password that is at least 8 characters: Programmer
Your user name is whizkid and your password is programmer
```

Comprehensive Lab WordGuess

In this case study, a word guessing game will be created. The word guessing game allows the player to guess the letters of a secret word. At the start of the game, the player is shown only how many letters the word contains through a set of dashes. When a letter matching one in the word is guessed, it replaces the appropriate dash. Play continues until the entire word is guessed letter-by-letter or when the player chooses to guess the entire word.

WordGuess Specification

WordGuess is played between the computer and a single player. The secret word is BRAIN. At the start of the game, six dashes are displayed (-----), one for each letter of the word. The player is repeatedly prompted for a letter guess. When a letter matching one in the word is guessed, the letter replaces the corresponding dash. Letters may be entered as uppercase or lowercase. However, only uppercase letters should be displayed. If the player enters an exclamation point (!), the player is prompted to guess the word. At that point the player either wins (a correct guess) or loses (an incorrect guess). Alternatively, the player can continue to guess letters until the entire word is revealed. The game ends by showing the player the total number of guesses.

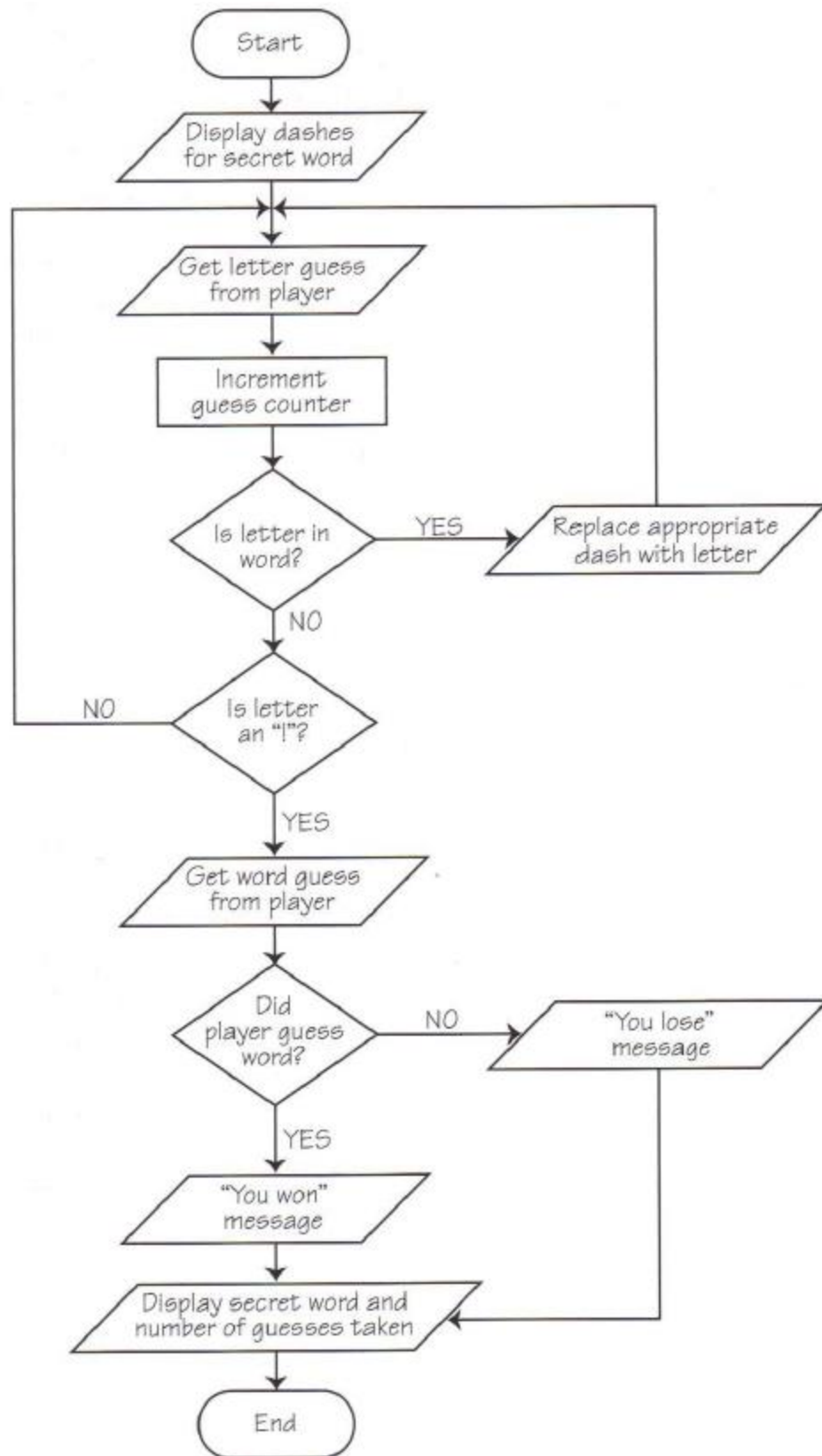
The WordGuess interface should display a row of dashes, one dash for each letter in the word. Prompts should be used to get letter guesses from the player. As corresponding letters are guessed, the letter is displayed instead of the dash. At the end of the game, the user should be shown the word along with the number of guesses.

The WordGuess output sketch:

```
WordGuess game.  
-----  
Enter a letter (! to guess entire word): a  
--A--  
Enter a letter (! to guess entire word): v  
--A--  
Enter a letter (! to guess entire word): !  
--A--  
  
What is your guess?  
brain  
You won!  
The secret word is BRAIN  
You made 3 guesses.
```

The WordGuess algorithm:

1. Display a row of dashes to represent the word.
2. Prompt the user for a letter guess.
3. If the letter guessed is part of the word, then display that letter in place of the corresponding dash.
4. Repeat steps 2 and 3 until all the letters have been guessed or an exclamation point has been entered by the user.
5. If an exclamation point has been entered, prompt the user to guess the entire word.
6. If the player correctly guesses the entire word or all the letters have been guessed, then display a message indicating that the player has won, otherwise the message should indicate that the player has lost.
7. Display the secret word and the number of guesses.



The input for WordGuess are letters typed by the user. A String variable to store the player's letter guess will be needed. The player may also choose to guess the entire word, so another String variable to store a word guess will be needed.

The output for WordGuess is the secret word displayed as dashes and then redisplayed whenever a dash should be replaced by a letter, a prompt for the user's guess, a message indicating a win or loss, and a message indicating the secret word and the number of guesses made.

Data generated by WordGuess is a count of the number of guesses made by the user. A counter variable to store the number of guesses will be needed. Other data used in this application can be represented by String constants to represent the secret word (SECRET_WORD) and to use as a loop sentinel (FLAG). A String variable will also be needed to "build" a new string that contains any correctly guessed letters.

Based on the algorithm and flowchart, the code design for the WordGuess application will include a loop structure to compare the user's input to the letters in the secret word. The best loop structure will be a `do-while` because under any condition the loop statements should iterate at least once. It also unknown how many times the loop statements should iterate, so a type of `while` statement, not a `for` statement, should be used. This comparison will be done with an `if` statement. A pseudocode algorithm for WordGuess follows:

```
Generate and display a set of dashes that represent the word
do
    update guesses counter
    Prompt user for a letter
    Convert to all uppercase
    Determine if letter is in word
    if letter is in word
        Create new string that contains the guessed letter
    while (all letters haven't been guessed and user hasn't chosen
to guess the entire word)
    if (! has been entered)
        get a word guess from player
        convert word to all uppercase
    if (word guessed equals secret word OR all the letters have
been guessed)
        display message that player has won
    else
        display message that player has lost
Display secret word
Display number of guesses
```

Unit 5 Labs

Lab GCD

Create a GCD application that prompts the user for two non-negative integers and then displays the greatest common divisor (GCD) of the two numbers. The GCD is the largest integer that divides into both numbers evenly. An algorithm for finding the GCD is called Euclid's Algorithm. Use the following pseudocode when implementing the GCD code:

```
while (num2 > 0) {  
    temp = num1 % num2;  
    num1 = num2;  
    num2 = temp;  
}
```

Application output should look similar to:

```
Enter a number: 32  
Enter a second number: 40  
The GCD is 8
```

Lab Password

Create a Password application that stores a secret password of your choice. The Password application should prompt the user for the password and then display "Welcome" if the correct password is typed. If after three tries the correct password has not been entered, the message "Access denied." should be displayed. Application output should look similar to:

```
Enter the password: homework  
The password you typed is incorrect.  
Enter the password: football  
The password you typed is incorrect.  
Enter the password: fish  
The password you typed is incorrect.  
Access denied.
```

Lab DefectiveCars

An auto company produced some models of cars that may be difficult to drive because the car wheels are not exactly round. Cars with model numbers 119, 179, 189 through 195, 221, and 780 have been found to have this defect. Create a CarRecall application that prompts a customer for the model number of their car to find out if it is defective and then displays "Your car is not defective." when the user typed a model number without a defect. Otherwise, the message "Your car is defective. It must be repaired." should be displayed. Application output should look similar to:

```
Enter the car's model number: 191
Your car is defective. It must be repaired.
```

Modify this app to allow the user to input as many model numbers as needed. Use 0 as a sentinel to end input.

```
Enter the car's model number or 0 to quit: 221
Your car is defective. It must be repaired.
Enter the car's model number or 0 to quit: 123
Your car is not defective.
Enter the car's model number or 0 to quit: 780
Your car is defective. It must be repaired.
Enter the car's model number or 0 to quit: 0
```

Lab CountVowels

Create a CountVowels application that prompts the user for a string and then displays a count of the number of vowels in the string. Application output should look similar to:

```
Enter text: Java Programming Assignment
The number of vowels in Java Programming Assignment is 8
```

There may be more labs posted on blackboard to assist your learning.

Harry Potter

```
"I solemnly swear that I am up to no good."
map.open();
```

```
"Mischief managed."
map.close();
```


GUI Practice Problems 2

Problem 2-1. Random Number Problem. Build an application where each time a button is clicked, a random number from 1 to 100 is displayed.

Problem 2-2. Price Problem. The neighborhood children built a lemonade stand. The hotter it is, the more they can charge. Build an application that produces the selling price, based on temperature:

Temperature	Price
<50	Don't bother
50 - 60	20 Cents
61 - 70	25 Cents
71 - 80	30 Cents
81 - 85	40 Cents
86 - 90	50 Cents
91 - 95	55 Cents
96 - 100	65 Cents
>100	75 Cents

Problem 2-3. Odd Integers Problem. Build an application that adds consecutive odd integers (starting with one) until the sum exceeds a target value. Display the sum and how many integers were added.

Problem 2-4. Pennies Problem. Here's an old problem. Today, I'll give you a penny. Tomorrow, I'll give you two pennies. I'll keep doubling the amount I'll give you for 30 days. How much will you have at the end of the month (better use a **long** integer type to keep track)?

Problem 2-5. Code Problem. Build an application with a text field and two buttons. Type a word or words in the text field. Click one of the buttons. Subtract one from the Unicode value for each character in the typed word(s), then redisplay it. This is a simple encoding technique. When you click the other button, reverse the process to decode the word.