

Patchogue-Medford High School

**Introduction to Computer
Programming:**

C++ Notes and Projects

Ms. Moshman

Student Name _____

Period _____

Student Name: _____

Project Name	Date Due	Points	Grade (If there is no grade, then you did not show me the project!)
Scavenger Hunt			
Initials			
Conversion			
Prj1A			
Prj1B			
Prj1C			
Prj1D			
Accumulators			
Telegram			
Select Statement			
Angles			
Planets			
Rock Paper Scissors			
For Loops			
Mini Project (RPS)			
Dowhile			
Dowhile1			
Dice			
Guess			
Hilo7			
Old MacDonald			

C++ -- A Background

C++ was developed at Bell Labs in 1972 as a high level language. Its form is different from LOGO and the commands will be different.

When you load C++ you must set up an environment as to where your programs will be stored and what libraries will be used. C++ uses pre-written programs to help it do its work. You will write your code and then it will be compiled or turned into machine language. The code you might see sometimes is Hexadecimal (0 – 15) but since there are only 10 digits, the letters a, b, c, d, e, f are used. A Hexadecimal number might be 37, 4A, A7, FF.

37 means $3 * 16 + 7$ or 55

4a means $4 * 16 + 10$ or 74

A4 means $10 * 16 + 4$ or 164

FF means $15 * 16 + 15$ or 255.

Each number means something to the computer's CPU. So if you see a bunch of A5 5F ... you have found the machine language section of C++.

All files must be set up in a specific way in order to run correctly. The programming will be input – process – output.

Input – getting data into the memory

Process – what to do with the data (output, decision, store)

Output – Get data to the screen, printer, file

C++ -- Notes

Approximately 5 volts of electricity is traveling through the computer at any 1 time. Inside there are chips containing tiny transistors or switches. The switch has 2 states: on and off. Most Pentium chips contain over 3 million transistors. The state of the switch corresponds to a number (0 or 1) which is the basis for the binary number system. One switch is called a bit and 8 switches or 1,048,576 bytes. Each switch in a byte corresponds to a power of 2:

128 64 32 16 8 4 2 1

If all the switches are off the value of the byte = 0. If all the switches are on, the value of the byte is 255.

Ex:

0 0 0 1 0 1 0 0 = 20

0 0 1 1 0 0 1 1 = 51

Each keystroke or combination of keystrokes sets switches and sends a value into RAM.

When you write programs involving numbers, you must tell the computer what types of numbers you will be using in your calculations.

Integers are represented using 2 bytes → 0 to 2^{16} . The largest and smallest values range from -32768 to 32767.

Real numbers are represented using 4 bytes → 0 to 2^{32}

The Path of a C++ Program:

You type code into an editor. This becomes the source code xxx.CPP. The code is then compiled and changed into an object code xxx.OBJ. The object code is then linked with library functions to generate an executable code or xxx.EXE which is then run.

When the code is compiled, it is turned into HEXADECIMAL or base 16. The digits range from 0 to 9 then the letters A, B, to F for 10 – 16

Ex:

3D means $3 * 16^2 + 13$

FF means $15 * 16 + 15$

21AB means $2 * 16^3 + 1 * 16^2 + 10 + 11 = 8469$

Programming:

A **program** is a series of instructions to make the computer do something.

→ The instructions MUST BE ORDERLY!!

Example:

To sharpen a pencil:

1. Put pencil in sharpener
2. turn handle (loop)
3. check point
4. remove pencil

To Wash a Car:

1. Wet
2. Soap
3. Rinse
4. Dry

C++ Scavenger Hunt

Directions: Open up notepad (Start-programs-notepad), put your name, date, period at the top. Type in 1. (and then the answer). Copy and paste the web address after the answer to each question. Print this out when you are done.

1. Who is the father of C++?
2. What year was C++ created?
3. What country is the father of C++ from?
4. Who/what company developed the original C? What year?
5. Define Object Oriented Programming.
6. What is the difference between C and C++?
7. What is Java?
8. When, where and who developed Java?
9. What brand and version of C++ is on the computer in front of you?
10. How do you write a comment in C++? (there are two ways.. give them both)
11. Define SOURCE code.
12. Define OBJECT code.
13. What is the difference between and INTEGER and a REAL value?
14. How many bits are in a byte?
15. Name three C++ tutorial sites on the internet.
16. Name 3 Java tutorial sites on the internet.
17. What is an INFORMATION SYSTEMS degree? Name 3 colleges with this major.
18. What is a COMPUTER SCIENCE degree? Name 3 colleges with this major.
19. What is a COMPUTER ENGINEERING DEGREE? Name 3 colleges with this major.

Name _____

Date _____

Moshman- Programming in C++

When starting OR exiting a program in C++, follow these directions EVERY TIME!!!

To start a new program:

1. Open Borland C++
2. File
3. New – Project
4. Name : h:\cpp\filename.ide
5. Change Target type to **Easywin**
6. Once Easywin is selected, **Class Library** should be checked
7. Click OK

When you are finished with your project:

1. Go to File
2. Save all
3. Project
4. Close Project
5. Then you may exit the program.

The text of a program is governed by rigid rules of syntax and style. The syntax is checked by the compiler, which does not allow a program with syntax errors to compile.

Comments complement the code, document functions, and explain obscure places in the code. The symbols are // for a single line comment... or /* */ for a group of commented code. Comments can also be used to “comment out” (temporarily disable) statements in the program.

Reserved Words and Programmer-Defined Names

The text of the program contains **reserved words**, which are used for a particular purpose in the language, as well as some names determined by the programmer. C++ is **case-sensitive**, so all words must be spelled with uppercase and lowercase letters rendered correctly. All *reserve words* use lowercase letters.

A partial list of the C++ reserved words is shown below:

char	sizeof	if	default
int	typedef	else	goto
float	const	for	return
double	static	while	class
short	void	do	private
long	enum	switch	public
unsigned	struct	continue	protected
signed	union	break	

What does a program consist of?

A program consists of functions and must have a function called **main()**, which gets control first. Functions must be declared before they can be used. This is accomplished by placing **function prototypes** somewhere near the top of the program. The actual function code is called the **body** of the function. It is placed inside braces in the function **definition**.

A programmer gives names to functions, variables, and constants, trying to choose names that make the program more readable. **Names may contain letters, digits, and the underscore character. They cannot start with a digit.**

C++ provides a vast collection of preprogrammed functions and classes, grouped in the standard library and in the class libraries. Prototypes for library functions and class functions and class definitions are contained in **header files** which are included into your program using the **#include** preprocessor directive.

Input and Output:

Input and Output is implemented by means of **stream I/O classes**. A **class** is a key C++ concept and many elements of the C++ syntax are associated with defining and using classes.

****NOTE****

The C++ stream I/O class library provides two ready-to-use classes: one for input, called *standard input*, and one for output, called *standard output*. We can think of standard input as the keyboard and standard output as the computer screen. The name of the standard input in programs is *cin* and the name of the standard output is *cout*.

cin and **cout** are defined in the header file **iostream.h** which is normally included at the top of the program:

```
#include <iostream.h>
```

The output operation:

- **inserts** data into an output stream.
- **Stream insertion** operator is denoted by the << symbol
- **cout<<**
- shows a **prompt**

The input operation:

- **extracts** data from an input stream.
- **Stream extraction** operator is denoted by the >> symbol
- **cin>>**

```
example:   cout<< "Enter a letter"<<endl;  
           cin>>letter;                //Reads one letter or integer
```

****NOTE****

Program code consists mostly of declarations and executable statements, which are normally terminated with a semicolon.

C++ syntax is not very forgiving and may frustrate a beginner – there is **NO SUCH THING AS “JUST A SEMICOLON”!**

```
//Name
//Date
//Period
//Simple Output
//Project name: output1.ide
```

```
/* This is an example of your first program in C++. Everything between the frontslash *
and * frontslash is a comment and does not affect execution of the program */
```

```
/* Any programs handed in should have your name, period, and date in comments right
at the top of the program as shown above. */
```

```
//The program begins on the next line, the two slashes on a line mean that
//everything that follows is a comment. NO slashes are needed at the end of
//the code... just at the beginning of the comment.
```

```
# include <iostream.h>    //This provides access to the header(library) file
                        //used for output
```

```
int main()    //The compiler looks for int main to start executing statements.
{
```

```
    cout<<"Hello, this is my first C++ program!"<<endl;
```

```
return 0;    //Every int main must have a return 0; to return control to the
}            //operating system.
```

```
//Hit F7 to compile your program
```

```
//Hit F5 to execute your program
```

Complete the following to your program:

- ```
/*
1) Insert a line after the cout to output your name.
2) Remove the <<endl from the cout<<"Hello line, what happens when you
 run the program? (keep the ;)
3) Put the endl back in and have the program print out your name, address and
 phone number on separate lines.
4) Put this line before the other couts
 cout<<"This is another way to get to the next line\n";
5) Change the \n to \n\n\n\n --What happens?
6) Add the line cout<<"Check this\n\n\nout!!!";
 What happens?
7) Add the line cout<<"\tThis is a tab"<<endl;
8) Go to File-Open -browse to d: Then bc5 then include and you will see
 iostream.h. This is the prewritten code that was included
 with the package. You can see there are lots of library files.
 C++ is HUGE!!!
*/
```

# *Initials Project*

Please name this: **initials.ide**

**Goal: You are to use simple output commands to make your initials.**

**You must use ALL THREE initials!! (If you do not have a middle name, use your last initial twice.)**

```
CCCCCCCCCCCC M M M M
C M M M M M M M M
C M M M M M M M M
C M M M M M M M M
C M M M M M M
C M M M M
C M M M M
C M M M M
C M M M M
CCCCCCCCCCCC M M M M
```

Remember: You must put your name, date, and period at the top of each program.

**2. If you have time, you may work on the same project using PASCAL's triangle. Save as: pascal.ide**

## Variables:

Most programmers refer to memory locations using symbolic names called **variables**. The programmer gives his variables meaningful names that reflect their role in the program. The compiler takes care of all the details – allocation of memory space for the variables and representation of data in the computer memory.

The term **variable** is borrowed from algebra because, as in algebra, variables can assume different values and can be used in **expressions**.

In a computer program, variables are actively manipulated by the program. A variable may be compared to a slate on which the program can, from time to time, write a new value and from which it can read the current value.

Example:     **a = b + c;**

This does not represent an algebraic equality, but instead a set of instructions:

1.     get the current value of b;
2.     get the current value of c;
3.     add the two values;
4.     assign the result to a (write the result in a)

## Data Types:

C++ has the following eleven **built-in** types designated by reserved words:

|             |                |
|-------------|----------------|
| char        | unsigned char  |
| int         | unsigned int   |
| short       | unsigned short |
| long        | unsigned long  |
| float       |                |
| double      |                |
| long double |                |

**\*\*Because variables of different types occupy different amounts of memory space, we say that they have *different* sizes.**

WE WILL MOSTLY USE THE *char*, *int*, *float*, and *double data types*.

| TYPE          | SIZE         | USE                                                                                                     |
|---------------|--------------|---------------------------------------------------------------------------------------------------------|
| <b>char</b>   | 1 byte       | One character, or a small integer in the range from 128 to 127                                          |
| <b>int</b>    | 2 or 4 bytes | An integer in the range from $-2^{15}$ to $2^{15} - 1$ or from $-2^{31}$ to $2^{31} - 1$ , respectively |
| <b>float</b>  | 4 bytes      | A real number in floating point representation ( <b>decimal</b> )                                       |
| <b>double</b> | 8 bytes      | A double-precision real number in floating point representation.                                        |

### **Constants:**

Constants are sometimes used in a C++ program. The most important reason for using constants is easier program maintenance. If the program is modified in the future and the value of a constant has to be changed, only the constant declaration has to be changed by the programmer.

Examples:

```
const double hamburgerPrice = 1.19;
const double cheeseburgerPrice = hamburgerPrice + .20;
```

**OR**

```
const double hamburgerPrice = 1.19,
 cheeseburgerPrice = hamburgerPrice + .20;
```

```
const double taxRate = .08;
```

```
...
```

```
taxAmt = amt * taxRate;
```

## Conversion Program 1

Name Project: conversion.ide

### Convert

1. Feet to inches
2. Inches to feet and inches
3. Yards to inches
4. Inches to yards and inches
5. Feet to yards and feet
6. Yards to feet

Bonus: Convert Inches to Yards, feet and inches  
(if test 522 inches = 14 yds, 1 foot, 6 inches.)

Units:

12 inches = 1 foot

36 inches = 1 yard

3 feet = 1 yard

### Program

1. Do each conversion input and output
  - a. Get the data in, process it, output it
2. Test the program with the numbers below
  - a. Enter 2 feet and get 24 inches
  - b. Enter 26 inches get 2 feet 2 inches
  - c. Enter 2 yards and get 72 inches
  - d. Enter 42 inches and get 1 yard 6 inches
  - e. Enter 5 feet and get 1 yard 2 feet
  - f. Enter 2 yards and get 6 feet

Try the conversion with pencil and paper and teach the computer how to do it.

Remember % is the remainder function

$16 \% 5 = 1$

16 divided by 5 = 3 remainder 1

# My Project Won't Work

## Errors:

1. `cout << "Enter a number" << endl`  
`cin >> x ;`
2. `cout << x << endl ;`  
`x = 3.5 + 4 ;`
3. `cout << setiosflags(ios::showpoint | ios::fixed) << setprecision (3) ;`  
`cout << " x = " << x ;`
4. `if x < 3 ;`  
`cout << x ;`  
`cin >> y ;`  
`cout << x + y ;`
5. `for (x = 1 ; x < 10 ; x ++ ) ;`
6. `cout " x = x " << endl ;`

## Developing algorithms

X is 5 years older than Y

Y is 13 - X is

Y is 23 - X is

Y is 33 - X is

Y is A - X is

X is 5 years younger than Y

Y is 13 - X is

Y is 23 - X is

Y is 33 - X is

Y is A - X is

How many cents in \_\_\_ nickels

3 nickels

5 nickels

x nickels

How many cents in \_\_\_ nickels and \_\_\_ dimes

3 nickels and 2 dimes

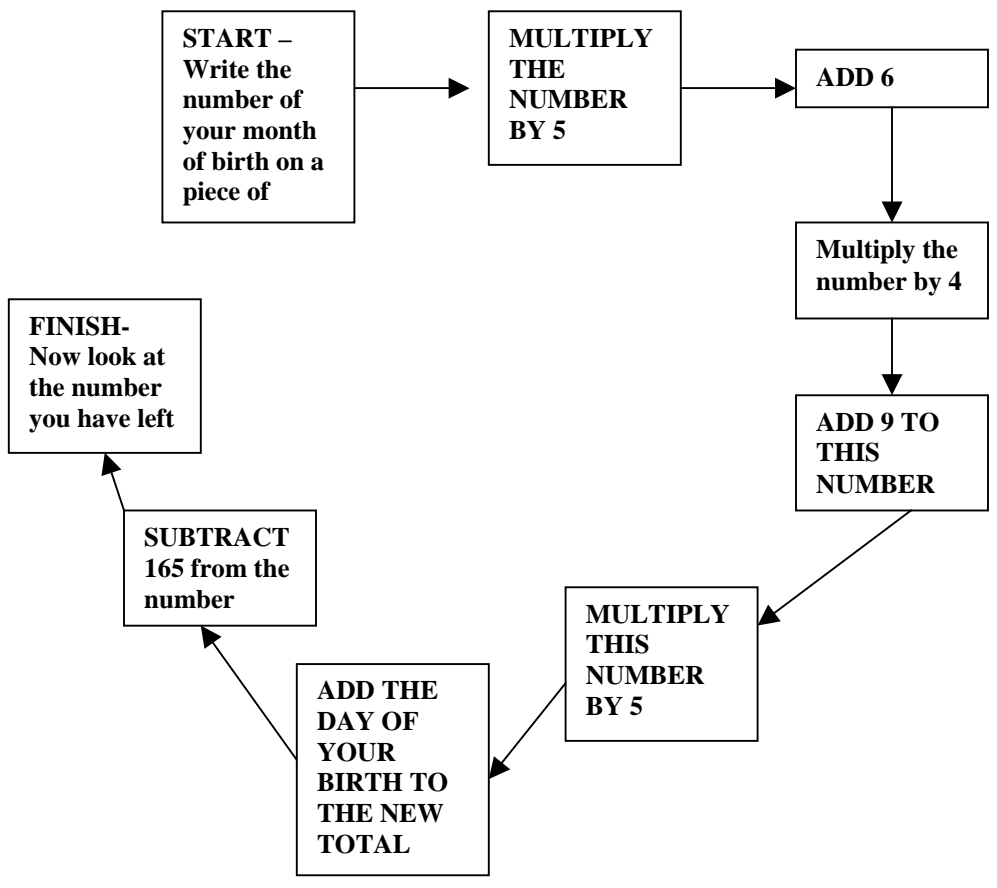
5 nickels and 5 dimes

x nickels and y dimes

Project #1A - Assigned :

DUE:

**NAME PROJECT: prj1a.ide**



\*\*\* the 2 digits tell you the day of your birth. The remaining digit(s) tells you the month of your birth.\*\*

Your assignment:

Write a program that will get you from start to finish with only two inputs from the user.

Hint: what information would you need to know from the person next to you to do this on your calculator?

## Projects 1B, 1C, 1D.

Assigned: \_\_\_\_\_  
Due: \_\_\_\_\_

Programs using int, long, float, cin, cout, expressions, calculations and iomanip.h formatting functions.

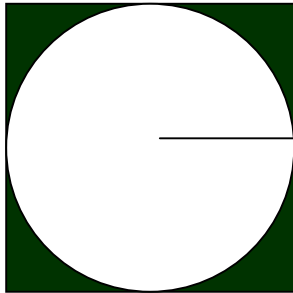
Prj1b → Ask a user for the year of his/her birth, and his/her age. Then double the birth year, add five, multiply by fifty, add the age, subtract 250 and divide by 100. Write the answer out with two digit of decimal accuracy.

Hint: use floats

Prj1c → Write a program to carry out the following chain of calculations: Begin by entering a number. Multiply it by 5, add 6 to the product, multiply by 4, add 9, multiply by 5. Now, cancel the last two digits of the final number, and subtract 1. What have you got?

Hint: use ints

Prj1d →



If you have a circle inscribed in a square and the only information given by the user is the radius of the circle. Write a program that asks for the radius of the circle then calculates the total area of the four corner pieces between the circle and the square. (The shaded region!) Use  $\pi = 3.14159$

Round to the nearest hundredth.

Hint: use floats

## Arithmetic Expressions:

Arithmetic expressions are written the same way as in algebra and may include literal and symbolic constants, variables, the arithmetic operators +, -, \*, and /, as well as parentheses.

### Compound Assignment Operators:

C++ has convenient shortcuts for combining arithmetic operations with assignment. The following table summarizes the **compound assignment** operators:

| Compound Assignment: | Is the same as:         |
|----------------------|-------------------------|
| <code>a +=b;</code>  | <code>a = a + b;</code> |
| <code>a -= b;</code> | <code>a = a - b;</code> |
| <code>a *= b;</code> | <code>a = a * b;</code> |
| <code>a /=b;</code>  | <code>a = a / b;</code> |

Example:

The following statement:

```
amt = amt + taxAmt;
```

Can be rewritten as:

```
amt +=taxAmount;
```

The second form may seem confusing at first, but once you get used to it, it becomes easier to use... it is more concise, and also emphasizes the fact that the same variable is being modified.

### Increment/Decrement Operators:

C++ has special **increment/decrement** operators (one of which gave the language its name). These operators are used as shorthand for incrementing and decrementing an integer value:

| Increment/Decrement: | Is the same as:         |
|----------------------|-------------------------|
| <code>a++;</code>    | <code>a = a + 1;</code> |
| <code>a--;</code>    | <code>a = a - 1;</code> |
| <code>++a;</code>    | <code>a = a + 1;</code> |
| <code>--a;</code>    | <code>a = a - 1;</code> |

We use the first two (increment values ) at our level:

### The Modulo Division Operator:

In addition to four arithmetic +, -, \*, /, C++ has the **% operator** for integers:

`a % b` → Which is “**a modulo b**” and means the **remainder** when a is divided by b.

Example:

**31 % 7 is equal to 3**

**365 % 7 is 1**

**\*\*Name this page: accumulators.ide**  
**\*\*Type in answers where the comments (//) are**

```
//Name
//Date
//"accumulators.ide"

#include <iostream.h>
#include <iomanip.h> // What is this used for? Do we need it here?
#include <math.h> // What is this used for? Do we need it here?

int main()
{

int num1=4, num2=10, num3=20, num4=10;

 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;
 num1=num1+1;
 num2=num2+1;
 num3=num3+1;
 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

 //Write down what happens

 num1++;
 num2++;
 num3++;
 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

 //Write down what happens to the numbers.

 num1--;
 num2--;
 num3--;
 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

 //Write down what happens to the numbers.

 num1=num1+num4;
 num2=num2+num4;
 num3=num3+num4;
 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

 //Write down what happens to the numbers.

 num1+=num4;
 num2+=num4;
 num3+=num4;
 cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

 //Write down what happens to the numbers.
```

```

num4=2;
num1*=num4;
num2*=num4;
num3*=num4;
cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

//What happens now?

num1-=5;
num2-=5;
num3-=5;
cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

//What happens to the numbers?

num1/=3;
num2/=3;
num3/=3;
cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

//What happens to the numbers
//What do we need to do?

num1%=5;
num2%=5;
num3%=5;
cout<<"Num 1 is "<<num1<<" Num 2 is "<<num2<<" Num 3 is "<<num3<<endl;

//What happens?
//What does the % function do?

cout<<"Enter a number."<<endl;
cin>>num1;
num2=num1*5;
num2+=6;

cout<<"Num 1 is "<<num1<<endl;
cout<<"The Answer Is "<<num2<<endl;

return 0;
}

```

# Logical Expressions and if-else statements:

The general form of the **if-else** statement in C++ is:

```
if (condition)
 statement 1;
else
 statement 2;
```

Where **condition** is a logical expression and *statement1* and *statement2* are either simple statements or **compound statements** (blocks surrounded by braces). The *else* clause is optional, so the *if* statement can be used by itself:

```
If (condition)
 Statement1;
```

When an *if-else* statement is executed, the program evaluates the condition and then executes statement1 if the condition is true, and statement2 if the condition is false. When *if* is coded without the *else*, the program evaluates the condition and executes statement1 if the condition is true. If the condition is false, the program skips *statement1*.

## Relational Operators:

C++ recognizes six relational operators:

| Operator | Meaning                     |
|----------|-----------------------------|
| >        | Is greater than             |
| <        | Is less than                |
| >=       | Is greater than or equal to |
| <=       | Is less than or equal to    |
| ==       | Is equal                    |
| !=       | Is not equal                |

### **\*\*NOTE\*\***

In C++ the "is equal" condition is expressed by the "==" operator, while a single "=" means assignment. Inadvertently writing = instead of == renders your conditional statement worse than meaningless. without generating a syntax error.

## Logical Operators:

C++ has two binary logical operators, "and" and "or", and one unary logical operator "not".

| Operator | Meaning |
|----------|---------|
| &&       | and     |
|          | or      |
| !        | not     |

## The expression:

condition1 && condition2

**Is true if and only if both condition1 and condition2 are true.**

condition1 || condition2

**Is true if either condition1 or condition2 or both are true.**

!condition1

**is true if and only if condition1 is false.**

C++

Name this program “conditional.ide”

```
// Name
//Date
//Conditionals If-then

#include <iostream.h>

int main()
{
 int n;
 cout<<"Testing for positive – negative or
zero."<<endl;
 cout<<"Enter a number to test."<<endl;
 cin>>n;
 cout<<endl;

 if (n<0) cout<<n<<" is
negative"<<endl;
 if (n>0) cout<<n<<" is
positive"<<endl;
 if (n==0) cout<<"The number is
zero."<<endl;

 cout<<endl;
 cout<<"Testing for odd and even"<<endl;
 cout<<"Enter a number"<<endl;
 cin>>n;

 if (n%2 == 0) cout<<n<<" is
even"<<endl;
 if (n%2 == 1) cout<<n<<" is
odd"<<endl;

return 0;
}
```

## Name this program “conditional1.ide”

```
//Name
//Date
//Period
//Conditionals

#include <iostream.h>

int main ()
{
int a=4, b=5, c=10;

if (a < b)
 cout<<"a is less than b"<<endl;
else
 cout<<"a is greater than b"<<endl;

if (b > 0)
 cout<<"b is positive"<<endl;
else
 cout<<"b is a negative
number."<<endl;

if (a==c)
 {cout<<"a is "<<a<<endl;
 cout<<"c is "<<c<<endl;
 cout<<"a equals c "<<endl;
 }

cout<<"Enter the number 5"<<endl;
cin>>a;
If (a==5)
 {cout<<"you listened "<<endl;
 cout<<"...yay for you"<<endl;
 }
else
{cout<<"you did not listen..."<<endl;
cout<<"...thanks for nothing."<<endl;
}
return 0;
}
```

## **Calculate the cost of a telegram**

**The cost to send a telegram is \$2.50 for the first 10 words or less and \$.15 for each additional word. Write a program to calculate the cost of sending a telegram under these conditions.**

Name this page: telegram.ide

Input – number of words

Process – calculate the cost of the telegram under the above conditions.

Output – the cost of the telegram

Plan:

Prompt for the number of words

Input the number of words

Calculate the cost -

1. make a decision of the number of words
2. up to 10 the cost is \$2.50
3. over 10 the cost is \$2.50 + .15 a word for each word over 10  
develop an algorithm

11 words –  $2.50 + 1 * .15$

12 words –  $2.50 + 2 * .15$

13 words –  $2.50 + 3 * .15$

x words –  $2.50 + ? * .15$  ← find the rule

Output – the cost of the telegram

Try your programs on the following:

7 words - \$2.50

15 words - \$3.25

# The Switch Statement

There are situations where a program must take one of several actions depending on the value of some variable or expression. Such situations often arise when the program is processing commands, events, menu choices, or transactions of different types.

If the program has to handle just **two** or **three** possible actions, you can easily use the **if-else** statement. However, if the number of possible actions is large, the use of if-else becomes inefficient. The **switch statement** can be used for such situations.

General Form:

```
Switch (expression)
{
 case A: //Take action A
 statementA1;
 statementA2;
 ...
 break;

 case B: //Take action B
 statementB1;
 statementB2;
 ...
 break;

 //Can have more cases from C – Y here.

 case Z: //Take action Z
 statementZ1;
 statementZ2;
 ...
 break;

 default: //Take some default action

 Break;
}
```

- A,B, ..., Z are integer or character **constants**. When a switch is compiled, the compiler creates a table of these values and the associated addresses of the corresponding “cases” (code).
- When the switch is executed, the program first evaluates expression to an integer. It then finds it in the table and jumps to the corresponding “case”.
- If the value is not in the table, the program jumps to **default**.
- The **break** statement at the end of each case tells the program to jump out of the *switch* and continue with the first statement after the switch.

## **\*\*NOTE\*\***

**Break** is used to break out of a loop or a switch. A *break* must be inside a loop or switch. In the case of nested loops or switches, a break tells the program to break out of the innermost loop or switch that contains it but does not affect the control flow in the outer switches or loops.

## SELECT STATEMENTS – ANOTHER WAY TO USE CONDITIONALS

*//Select Statement Worksheet – name this page select.ide*

*// The select statement can be used instead of nester if-else statements*

*// As you go through the three sections remember you can use /\* and \*/ to comment*

*// out many lines of code all at once.*

```
include <iostream.h>
```

```
int main()
```

```
{
 int count, rnum;
 cout<<"Please input a number from 1 to 3"<<endl;
 cin>>rnum;
 switch (rnum)
 {
 case 1:cout<<"You entered 1\n";
 cout<<"one"<<endl;
 break;
 case 2:cout<<"You entered 2\n";
 break;
 case 3:cout<<"You entered 3\n";
 break;
 }
 cout<<endl;
```

*// 1) add in case sections for the numbers 4 – 6*

*// 2) Enter a number other than 1 – 6, what happens???????????*

*// 3) after the last break add in the following*

```
 default:cout<<"You didn't enter a number from 1-6\n";
 break;
```

*// Type in another example of a switch statement*

```
int number;
cout<<"Enter the a number from 1 to 6\n";
cin>>number;
switch (number)
{
 case 1: case 3:case 5:
 cout<<"You entered an odd number\n";
 break;
 case 2:case 4:case 6:
 cout<<"You entered an even number\n";
 break;
}
cout<<endl;
```

*// Convert the following nested if else statement to a switch statement.*

```
Int grade;
```

```
Cout<<"Enter quiz grade from 14-20\n";
```

```
Cin>>grade;
```

```
if (grade==20)
```

```
 cout<<"Superior!"<<endl;
```

```
 else if (grade >=18)
```

```
 cout<<"Excellent!"<<endl;
```

```
 else if (grade >=16)
```

```
 cout<<"Good!"<<endl;
```

```
 else if (grade >=14)
```

```
 cout<<"Fine work"<<endl;
```

```
 else cout<<"Not so great!"<<endl;
```

```
return 0;
```

```
}
```

## Three Projects - Conditional & Switch Statements

### Programming in C++ - Conditional and Switch Statement Programming Assignment(s)

#### 1) angles.ide

An angle is given as a positive real number of degrees. Angles from  $0^\circ$  to (but not including)  $90^\circ$  fall in the first quadrant,  $90^\circ$  to  $180^\circ$  are in the second,  $180^\circ$  to  $270^\circ$  are in the third, and  $270^\circ$  to  $360^\circ$  are in the fourth quadrant. The series of quadrants starts all over again at  $360^\circ$ . Write a program that reads the value of an angle, prints out the angle, and the quadrant it falls in. (Remember... 0, 90, 180, 270, 360... are not in quadrants... they are on the axes!)

#### 2) planets.ide (you must use a switch statement for this project)

Relative gravities of most planets in the solar system, the Sun, and the Moon, are shown below. They let us find what a given weight on Earth would weigh on each body.

|               |               |                |
|---------------|---------------|----------------|
| Sun = 27.94   | Mercury = .37 | Venus = .88    |
| Moon = .17    | Mars = .38    | Jupiter = 2.64 |
| Saturn = 1.15 | Uranus = 1.17 | Neptune = 1.18 |

Write a program that lets a user find his/her weight on any of these. It should ask to enter the first letter (type char) of the planet or body wanted, and then, if there's an ambiguous situation (as in S or M) it should prompt for the second letter. In other words, a nested switch structure is required. Another possibility is to make each body correspond to a number and then have the user enter the number for the body desired. (ex sun = 1, Moon = 2, etc.)

#### 3) rps.ide

Write a program that plays the childrens' game Rock, Paper, Scissors. Two char inputs will represent the two players. There are only three rules: Scissors cut paper, Rock Crushes Scissors, and Paper covers Rock. Have the program prompt for the two players choices then determine whether player 1 or player 2 is the winner of that round.

# Iterative Statements: While, For, do-while

**Loops** or **iterative** statements tell the program to repeat a fragment of code several times or as long as a certain condition holds. Without iterations, a programmer would have to write separately every instruction executed by the computer, and computers are capable of executing millions of instructions per second. Instead, programmers can use fewer instructions which the computer will repeat numerous times.

C++ offers three convenient iterative statements: *while*, *for* and *do-while*. Strictly speaking, any iterative code can be implemented using only the *while* statement, but the other two add flexibility and make the code more concise and idiomatic.

## For Loops:

Is a shorthand for the *while loop* that combines initialization, condition, and increment in one statement.

General Form:

```
for(initialization; condition; increment)
{
 statement1;
 statement2;
}
```

- **Initialization** is a statement that is always executed once before the first pass through the loop.
- **Condition** is tested before each pass through the loop
- **Increment** is a statement executed at the end of each pass through the loop.

**Example:** returns the sum of all integers from 1 to n, if  $n \geq 1$ , and 0 otherwise.

```
int sum=0;
int i;

for (i=1; i<=n; i++)
{
 sum+=i;

 return sum;
}
```

## While Loops:

General Form:

```
While (condition)
{
 statement1;
 statement2;
}
```

- condition can be any arithmetic or logical expression.
- It is evaluated exactly the same way as an if statement
- Statements inside braces are called the **body**
- If the body consists of **only one** statement, the surrounding braces can be dropped

```
While (condition)
 statement1;
```

### **\*\*NOTE\*\***

It is important **NOT** to put a semicolon after the while(condition). With a semicolon, the body of the loop would be an empty statement, which would leave *statement1* completely out of the loop.

**Three elements must be present, in one form or another, with any WHILE loop: initialization, a test of the condition, and incrementing.**

1. Initialization: the variables tested in the *condition* must first be initialized to some values. For example: `int i = 1`
2. Testing: The condition is tested before each pass through the loop. If it is false, the body is not executed, iterations end, and the program continues with the next statement after the loop. If the condition is false at the very beginning, the body of the *while* loop is not executed at all.
3. Incrementing: At least one of the variables tested in the condition must change within the body of the loop. Otherwise, the loop will be repeated over and over and never stop, and your program will **hang**. (use `i++`)

## Do-While Loop:

The do-while loop differs from the *while* loop in that the condition is tested *after* the body of the loop. This assures that the program goes through the iteration at least once.

General Form:

```
do
{

}
while(condition);
```

← Semicolon is at the end of this loop

The program repeats the body of the loop as long as *condition* remains true. It is better always to keep the braces with this type of loop.

### **\*\*NOTE\*\***

**This is the type of loop you might want to use with a game or a program that asks the user if they would like to do that again?**

Date:

Name this program forloops2.ide

Loops: the “for” statement

The for statement has the form:

```
For(initial_value,test_condition,step)
{
 //code to execute inside loop
};
```

- *Initial\_value* sets up the initial value of the loop counter.
- *Test\_condition* this is the condition that is tested to see if the loop is executed again.
- *Step* this describes how the counter is changed on each execution of the loop. (x++ → goes up by 1)  
(x+=2 → goes up by 2)

#### EXAMPLE 1:

**// The following code adds together the numbers 1 through 10**

```
//This variable keeps the running total
int total=0;
//This loop adds the numbers 1 through 10 to the variable total
for (int x=1; x<11;x++)
{
 cout<<"The total is "<<x<<endl;
 total = total + x;
}
```

#### EXAMPLE 2:

1. Change example 1 to a comment by adding a */\** to the beginning and *\*/* to the end of it.
2. Make sure that you put `#include <iomanip.h>` at the beginning because we are going to set width (setw)
3. Set num, sum=0, sum2=0 as integers.
4. Write the for loop with a initial value of x=0, the test condition as x<=10, and the step as +1.  
(\*Please remember to set x as an integer in the for loop)
5. { cout<<setw(10)<<x<<endl;}
6. Then add in another line: (in the same brackets): cout<<"Hello"<<endl;}
7. Now take brackets away completely around the loop and see what happens.
8. Now add in x\*x to your cout<<setw(10)<<x<<setw(10)<<x\*x<<endl;
9. sum+=x;
10. sum2+=x\*x; (in loop brackets still)
11. Add a cout with the same column width that will find the sum of x and the sum of x\*x.  
→ cout<<setw(10)<<sum<<setw(10)<<sum2<<endl;

#### ANOTHER EXAMPLE OF A FOR LOOP (NESTED FOR LOOP)

```
for (int x=1; x<=num; x++) //can replace 5 with num
{
 for(int y=1; y<=num2; y++) //can replace 10 with num2
 {
 cout<<x;
 }
 cout<<endl;
}

return 0;
}
```

Name \_\_\_\_\_

Date \_\_\_\_\_

***For - Loops Programming Assignment***

**Project Name : "forloops.ide"**

These are mini-projects... in between each project, use getch() and clrscr()

1. Write a for-loop that will find the sum of the odd numbers from 1 to 101.
2. Write code that will ask the user for an integer from 1 - 100. Use that number to write a for-loop that will print "I LOVE FRIDAYS!" that many times.
3. Write code that will make the following table (A for-loop must be used)

| Number | Number*10 |
|--------|-----------|
| 1      | 10        |
| 2      | 20        |
| 3      | 30        |
| 4      | 40        |
| 5      | 50        |
| 6      | 60        |

\*\*\* extra - find the sum of both columns and print it out

\*\*\* extra - ask for start and end values for the number that will be used in the chart.

4. Write code that will ask for a number from 5 - 10. Use a for-loop to print out a box of stars with that number for it's sides.

\*\*\*

\*\*\*

\*\*\* This box would be an example of 3 X 3

5. Write the code that will ask for two integers **start** and **end**. (**start** should be less than **end**.) Write a for-loop that will find a sum of the squares of the numbers from **start** to **end**.

Example - **start** is 3 ... **end** is 5...

Computer print out... **Sum is  $3^2 + 4^2 + 5^2 = 50$**

|                    |                           |
|--------------------|---------------------------|
| <b>Started on:</b> | <b>Name: "while1.ide"</b> |
| <b>Completed:</b>  |                           |

```
//Name
//Date -
//Period
//"while1.ide"

#include <iostream.h>
#include <conio.h> //used for getch and clrscr
#include <ctype.h> //used for toupper

int main()
{
cout<<"Simple while loop example "<<endl;

int a = 1;

while (a<5)
{
 cout<<a<<endl;
 a++;
}
 getch();
 clrscr();

//While loop that prints all even numbers between 11 and 23.
//Checked each time after .
//If condition is true will execute it again.

cout<<"Using a while loop to print the numbers 2 - 22 even."<<endl;

 int b = 12; //This variable holds the current number.

while(b <23)
{
 cout<<b<<endl;
 b+=2;
}
cout<<"all done"<<endl;
cout<<" "<<endl;
cout<<"Press enter to continue."<<endl;
getch();
clrscr();
```

(Continued on next page)

While1.ide

```
cout<<"Using a while loop to compare two numbers."<<endl;
```

```
int num1, num2;
char choice;
num1=0;
num2=10;
while (num1<num2)
{
 cout<<"num1 is "<<num1<<" num2 is "<<num2<<endl;
 num1++;
 getch();//add later
 //have students move the getch(); to the outside of the loop
 //What happens?
}
clrscr();
```

```
cout<<"Same program using the do while loop."<<endl;
```

```
cout<<"Testing Do While Loop"<<endl;
num1=0;
num2=10;
do
{
 cout<<"num1 is "<<num1<<" num2 is "<<num2<<endl;
 num1++;
}
while (num1<num2);
getch();
clrscr();
```

```
cout<<"Testing choice loop"<<endl;
do
{
 cout<<"Today is Monday"<<endl;
 cout<<"Do you want to write this again? Y/N"<<endl;
 cin>>choice;
 choice=toupper (choice);
}
while (choice=='Y'); //This choice must be capital.
```

**MINI PROJECT: Go back to the Rock paper scissors program and loop it to run again if the player types in Y.**

```
return 0;
}
```

**Assigned:**

**Due:**

```
//Name
//Date
//name this page dowhile.ide

include <iostream.h>

int main()
{
int count=0; //This means that you can initialize a variable upon declaring it.
int num1, a, b;
```

**//Loop #1 - Give the output on this sheet**

```
a=0;
b=5;
while (a<=b)
{
 cout<<"Hello"<<a<<endl;
 a++;
}
```

**//Loop #2 - Give the output on this sheet**

```
a=0;
b=5;
do
{
 cout<<"Hello"<<a<<endl;
 a++;
}
while (a<=b);
```

**// Loop #3 -**

```
int grade, numgrades;
cout<<"How many grades are you going to enter??\n";
cin>>numgrades;

count=0;
while (count<=numgrades)
{
 cout<<"Enter next grade\n";
 cin>>grade;
 count++;
}
```

**//did the code work as indicated? What simple change will fix it?**

Do While Loops

**//Loop #4**

```
cout<<"Enter an integer (Enter 0 to end)"<<endl;
cin>>num1;
cout<<"your number is "<<num1<<endl;
count=0;
while (num1 !=0)
 { //While loop
 count+=1;
 cout<<"Enter an integer (ENTER 0 TO END)"<<endl;
 cin>>num1;
 if (num1 !=0)
 cout<<"Your number is "<<num1<<endl;
 } //WHILE LOOP
cout<<endl;
cout<<"You entered "<<count<<" numbers."<<endl;
```

**// Loop #5**

**// now lets write the same code as a do- while loop.**

```
do
{
 cout<<"Enter an integer (ENTER 0 TO END)"<<endl;
 cin>>num1;
 count+=1;
 if (num1!=0)
 cout<<"Your number is "<<num1<<endl;
} //WHILE LOOP
while (num1 !=0);
cout<<endl;
cout<<"You entered "<<count<<" numbers."<<endl;
```

**//Are the numbers counted correctly in the above loop???**  
**//What change could be made to count them correctly???**

```
return 0;
}
```

**//Assignment – name this “dowhile1.ide”**

**// 1) Write a While- Do Loop to find the sum of the numbers from 1 to 100.**

**// 2) Write a Do-While loop to do the same as #1.**

**// 3) Write a While-Do loop that keeps reading in an integer and prints**

**// "I can't wait for summer!" Until the integer entered is a -1.**

**// 4) Write a Do- While loop to do the same thing.**

**// 5) Use a while loop to print the following table.**

```
// X X-Squared
// _____
// 0 0
// 1 1
// 2 4
// 3 9
// 4 16
// 5 25
```

**// 6) Change the above around so that you can input the start and end values  
// for the table.**

## C++ Review

### Include:

iostream -- necessary for cout and cin  
conio – necessary for clrscr() and getch()  
apstring – necessary for apstring  
iomanip – necessary for setprecision

### Variables:

Used to hold data within the program

Int – used for integer variables between  $-32767$  and  $+32768$   
Long – used for integers up to  $2^{32} - 1$   
Double -- used when the variable could be a fraction or decimal  
Apstring -- used when text is necessary

**Output:** when data or text is to be put on the screen

```
Cout<<"Text goes inside of quotes"
Cout<<"Text inside quotes and variables outside"<<variable
```

**Prompt:** a message to the user

```
Cout<<"Please enter a number then press the enter key"
```

**Input:** data entered into the program

```
Cin>>x;
```

**Decision:** if( ) what you want to happen

```
If(x < 5) cout<<"The number is less than 5"
If(x < 5)
{
 x++;
 y = x + y;
 cout<<"The numbers are "<<x<<" and "<<y;
}
```

**Loops:** for loop runs a fixed number of times

While loop allows the user to leave when a condition is met

```
For(x=1; x < 10; x++)
{
```

} ← x starts at 1, goes up by one, and ends when x is at 10. Everything inside the { } gets repeated as long as x is less than 10.

```
While (x < 10)
{
```

} ← As long as x remains less than 10 the loop runs.

Name \_\_\_\_\_

Date \_\_\_\_\_

### Quiz – C++ - Conditional Statements

1. Write an If-Else statement that will print “Hello” if num is positive and will print “goodbye” if num is negative.

2. Write a for-loop that will print the multiples of 4 from 4 to 40.

3. Write a **program** that will ask for two integers called num1 and num2 and output the greater of the two numbers.

4. Write a for loop that will calculate and print the sum of the numbers from 1 to 100.

For numbers 5 – 9, give the output for the code:

5.

```
sum=0;
for (count=1; count<=5; count++)
{
 if (count=1)
 {
 cout<<count;
 sum+=count;
 }
 else
 {
 cout<<"+"<<count;
 sum+=count;
 }
}
cout<<sum<<endl;
```

**OUTPUT**

6.

```
int num1, num2;

num1=20;
num2=3;
cout<<num1<<"*"<<num2<<" is "<< (num1*num2)<<endl;
cout<<num1<<"/"<<num2<<" is "<< (num1/num2)<<endl;
cout<<num1<<"%"<<num2<<" is "<< (num1%num2)<<endl;
```

**OUTPUT**

7.

```
int x=1;

while(x<=5)
{
 cout<<"I love C++"<<x<<endl;
 x++;
}
}
```

**OUTPUT**

8.

```
sum=0; x=1;

do
{
 sum=sum+x;
 x++;
 cout<<"Sum is... "<<sum<<" x is... "<<x<<endl;
}
while (x<=4);
```

**OUTPUT**

Date: \_\_\_\_\_

Name of Program: **DICE**

```
#include <iostream.h>
#include <conio.h> // for getch and clrscr
#include <ctype.h> // for toupper
#include <stdlib.h> // for random

int main()
{
 randomize(); // added later
 int die1,die2,totalroll;

 char choice;

 for (int x=1;x<=10;x++)
 {
 //die1=random(6); // This will give you random numbers from 0 to 5
 die1=random(6) + 1; // This will give you random numbers from 1 to 6
 cout<<"Dice rolled... "<<die1<<endl;
 }

 cout<<endl<<"rolling 2 dice"<<endl;
 for (int x = 1; x<=10;x++)
 {

 die1=random(6) +1;
 die2=random(6) +1;
 totalroll=die1+die2;
 cout<<"Die1... "<<die1<<" Die2... "<<die2<<" Totalroll... "<<totalroll<<endl;
 }
 cout<<"Press any key to continue."<<endl;
 getch();
 clrscr();

 cout<<endl<<"rolling 2 dice and counting how many nines"<<endl;

 int count=0; //Add this in -- (you must set count equal to zero if you are going to re
 roll each time)
 for (int x = 1; x<=10;x++)
 {

 die1=random(6) +1;
 die2=random(6) +1;
 totalroll=die1+die2;
 cout<<"Die1... "<<die1<<" Die2... "<<die2<<" Totalroll... "<<totalroll<<endl;
 //Leave the cout line above out if you are rolling the dice too much (100 times, etc)
 if (totalroll == 9)
 count++;
 }
 cout<<"There were "<<count<<" nines in 10 rolls "<<endl;

 return 0;
}
```

**PROJECT: Add in counters for how many of each roll of the dice for 600 rolls of the dice**

- a) 7's for 600 rolls of dice
- b) Add 2's
- c) 12's

# *Guess a Number*

Name page: `guess.ide`

**Design a program to make a guessing game.**

**Object of the game:      Guess a number from 1 to 100.**

1. You want to design your program to allow 10 guesses.
2. After each guess, you want to tell the player if the number is correct, too high, or too low.
3. If the number is too high, have the player guess another number.
4. Set integers for guess, number, count, and x.
5. After 10 guesses, ask the player if they would like to play again?

**FOLLOW THIS ORDER TO WRITE YOUR PROGRAM:**

1. `int guess=0, number, count=0, x;`
2. `char tryagain;`
3. Put in prompts that will tell you the object of the game, how many guesses, and what will happen after 10 guesses.
4. **Do loop**
  - a. `number=`
  - b. `while (count is less than 10 and guess is not equal to the number)`
  - c. `count++`
  - d. `prompt "Please enter your try #"<<count<<"`;
  - e. `cin>>guess;`
  - f. `if (guess is less than number)`
  - g. `prompt "too low"`
  - h. `else if (guess is greater than number)`
  - i. `prompt "too high"`
  - j. `else prompt<<"CONGRATS"`
  - k. `close your while loop`
  - l. `if (count == 10 and guess does not equal number)`
  - m. `{prompt "too bad. The number was"<<number<<"endl; }`
  - n. `prompt "Would you like to try again?"`;
  - o. `cin>>tryagain;`
  - p. `clear the screen`
5. Close do loop
6. `While (toupper(tryagain)=='Y');`

## Review for High-Low Seven:

### Name this hiloreview.ide

```
#include <iostream.h>
#include <stdlib.h> //for random
#include <conio.h> //for clrscr() and getch()
#include <ctype.h> //for toupper

int main()
{
 randomize();
 int die1, die2, totalroll;
 char choice;

 cout<<endl<<"rolling 2 dice"<<endl;

 for(int x=1; x<=10; x++)
 {
 die1 = random(6) + 1;
 die2 = random(6) + 1;
 totalroll = die1 + die2;
 cout<<"Die1... "<<die1<<" Die2... "<<die2<<" Total roll... "<<totalroll<<endl;
 }

 //STOP and run the above

 getch();
 clrscr();

 //entering a char value

 cout<<"Do you like Mondays?"<<endl;
 cin>>choice;
 choice=toupper(choice); //uppercases the input
 cout<<"Choice is "<<choice<<endl;

 //STOP and run the program

 int num;
 char exit;
 do
 {
 num=random(10);
 cout<<"Num is "<<num<<endl;
 if(choice == 'Y' && num < 5)
 cout<<"Hello"<<endl;
 else if(choice == 'N' && num > 5)
 cout<<"Goodbye"<<endl;
 else if (choice == 'Y' && num==5)
 cout<<"It is Friday "<<endl;

 cout<<"Do you want to go again? Press Y or N"<<endl;
 cin>>exit;
 }
 while (exit =='Y')

 return 0;
}
```

## High-Low-Seven Assignment

name page: hilo7.ide

In the game of High-Low-Seven a person places a bet of High, Low or Seven then rolls two dice. If a bet of high is placed a roll greater than 7 happens they are a winner. If a bet of Low is placed and a roll less than 7 happens then they are a winner. If they bet Seven and roll a 7 they are a winner. Any other combination of bets and rolls they lose.

- 1) Write a program to simulate the game of High-Low-Seven. Ask for the Bet (a char) 'H' or 'L' or 'S'. Use toupper from the ctype.h include file to convert the bet to uppercase (ex. – bet=toupper(bet)). Roll the dice.. get the total roll and use if-else statements to determine a win or a lose.
- 2) Make the game play multiple times

### **EXTRA CREDIT**

- 3) Add in an initial bank of \$500—ask for a monetary bet – play the game – add or subtract depending on the outcome.

# Strings

## Literal Strings:

A **literal string** is a string of characters in double quotes. The string may include “escape” characters such as ‘\t’ (tab), ‘\’ (double quote), ‘\n’ (newline), etc.

## The apstring class:

A typical string class allows a programmer to initialize a string class object (variable) with a literal string and to assign a value to a string class variable from a literal string.

Allows us to handle character strings the same way as we handle *int* or *double*.

The definition of the class and declarations of its functions and operators must be included into the program. As usual, they are placed in the header file, *apstring.h*.

The apstring class provides three ways to declare a string:

**#include “apstring.h”**

```
apstring str1; //Declares an empty string
apstring str2 = “Hello”; //Initialize a literal string
apstring str3 = str2; //Initialize to a previously defined string
```

## Old MacDonald – Functions Assignment

Write a program that generates the verses of the children’s song below. Don’t worry about the ungrammatical qualities inherent in the use of “a” and “an” in your attempt to writing the program. You should include a function with two parameters capable of generating any of the verses when the appropriate arguments are passed.

It is possible to write this with just one function or with many. Don’t get too complicated... but if you really feel you understand the function concept try to use more than one function.

Old MacDonald had a farm,  
Ee I ee I oh!  
And on his farm he had some chicks,  
Ee I ee I oh!  
With a cluck-cluck here,  
And a cluck-cluck there,  
Here a cluck, there a cluck  
Everywhere a cluck-cluck  
Old MacDonald had a farm,  
Ee I ee I oh!

Old MacDonald had a farm,  
Ee I ee I oh!  
And on his farm he had some cows,  
Ee I ee I oh!  
With a moo-moo here,  
And a moo-moo there,  
Here a moo, there a moo  
Everywhere a moo-moo  
Old MacDonald had a farm,  
Ee I ee I oh!

Old MacDonald had a farm,  
Ee I ee I oh!  
And on his farm he had some pigs,  
Ee I ee I oh!  
With a oink-oink here,  
And a oink-oink there,  
Here a oink, there a oink  
Everywhere a oink-oink  
Old MacDonald had a farm,  
Ee I ee I oh!

### Grading

75 -- Have “Old MacDonald” print out as given

80 -- Be able to put in the “object” and “sound” you want the song to sing.

90 -- Put in a loop to be able to choose the song again.

100 -- Put in another children song of your choice. Put in a prompt that asks Would you like to sing “Old MacDonald” or “your song choice”?

***Other Useful  
C++  
Information:***

# BGI Graphics in C++

*You need to start a new project in a different way:*

1. *Set target type to application.exe*
2. *Set Platform to DOS (Standard)*
3. *Target model is Large*
4. *Click the Class Library*
5. *Click the Emulation*
6. *Click BGI button.*

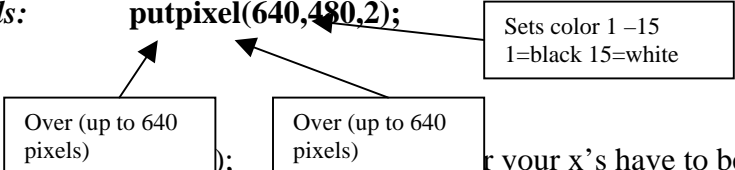

When you start your program, you must type this in every time when using graphics... so ***copy and paste!***

```
#include <iostream.h>
#include <graphics.h>
#include <math.h> //for fabs
#include <conio.h> //for getch()
#include <dos.h> //for delay
#include <stdlib.h> //for random

int main()
{
 int driver, mode, errorcode;
 driver=DETECT;
 initgraph(&driver, &mode, "d:\\bc5\\bgi");
 errorcode=graphresult();

 if(errorcode!=grOk)
 {
 cout<<"Error in BGI"<<endl;
 }
}
```

## ***To put different graphics on a page:***

1. ***Pixels:*** `putpixel(640,480,2);`  

2. ***Lines:*** `line(x1, y1, x2, y2, c);` for your x's have to be <640 and y's <480!!  


*There is no set color in the line( ). You have to put in: setcolor( ); before you draw the line in!!*

Colors from 1 - 15

ASCII stands for American Standard Code for Information Interchange.

Computers can only understand numbers, so an ASCII code is the numerical representation of a character such as 'a' or '@' or an action of some sort.

ASCII was developed a long time ago and now the non-printing characters are rarely used for their original purpose.

Below is the ASCII character table and this includes descriptions of the first 32 non-printing characters. ASCII was actually designed for use with teletypes and so the descriptions are somewhat obscure. If someone says they want your CV however in ASCII format, all this means is they want 'plain' text with no formatting such as tabs, bold or underscoring - the raw format that any computer can understand. This is usually so they can easily import the file into their own applications without issues. Notepad.exe creates ASCII text, or in MS Word you can save a file as 'text only'

| ASCII VALUE REFERENCE |                 |                       |              |        |       |       |       |       |
|-----------------------|-----------------|-----------------------|--------------|--------|-------|-------|-------|-------|
| LETTERS (Upper)       | LETTERS (Lower) | Numbers & Punctuation | Non-Standard |        |       |       |       |       |
| A=65                  | a=97            | 0=48                  | *=42         | ` = 96 | ž=158 | ˆ=184 | Ń=209 | ê=234 |
| B=66                  | b=98            | 1=49                  | + =43        | ¡=130  | ı=159 | ˙=185 | Ô=210 | ë=235 |
| C=67                  | c=99            | 2=50                  | ,=44         | ª=131  | ˚=160 | ˘=186 | Ó=211 | ì=236 |
| D=68                  | d=100           | 3=51                  | -=45         | «=132  | ı=161 | »=187 | Ô=212 | í=237 |
| E=69                  | e=101           | 4=52                  | .=46         | «=133  | ç=162 | ¼=188 | Õ=213 | î=238 |
| F=70                  | f=102           | 5=53                  | /=47         | «=134  | £=163 | ½=189 | Ö=214 | ï=239 |
| G=71                  | g=103           | 6=54                  | : =58        | «=135  | ¤=164 | ¾=190 | ×=215 | š=240 |
| H=72                  | h=104           | 7=55                  | ; =59        | «=136  | ¥=165 | ı=191 | Ø=216 | ř=241 |
| I=73                  | i=105           | 8=56                  | <=60         | «=137  | ¦=166 | À=192 | Ū=217 | ô=242 |
| J=74                  | j=106           | 9=57                  | = =61        | «=138  | §=167 | Á=193 | Ū=218 | ó=243 |
| K=75                  | k=107           |                       | >=62         | «=139  | ¨=168 | Â=194 | Ū=219 | ô=244 |
| L=76                  | l=108           |                       | ?=63         | «=140  | ©=169 | Ã=195 | Ū=220 | õ=245 |
| M=77                  | m=109           |                       | @=64         | «=142  | ª=170 | Ä=196 | Ū=221 | ö=246 |
| N=78                  | n=110           |                       | [ =91        | «=145  | «=171 | Å=197 | ƒ=222 | ÷=247 |
| O=79                  | o=111           |                       | \ =92        | «=146  | «=172 | Æ=198 | ß=223 | ø=248 |
| P=80                  | p=112           | !=33                  | ] =93        | «=147  | ©=174 | Ç=199 | à=224 | ù=249 |
| Q=81                  | q=113           | "=34                  | ^ =94        | «=148  | «=175 | È=200 | á=225 | ú=250 |
| R=82                  | r=114           | #=35                  | { =123       | «=149  | «=176 | É=201 | â=226 | û=251 |
| S=83                  | s=115           | \$=36                 | =124         | «=150  | ±=177 | Ê=202 | ã=227 | ü=252 |
| T=84                  | t=116           | %=37                  | } =125       | «=151  | ²=178 | Ë=203 | ä=228 | ý=253 |
| U=85                  | u=117           | &=38                  | ~ =126       | «=152  | ³=179 | Ì=204 | å=229 | þ=254 |
| V=86                  | v=118           | '=39                  |              | «=153  | ´=180 | Í=205 | æ=230 | ÿ=255 |
| W=87                  | w=119           | (=40                  |              | «=154  | µ=181 | Î=206 | ç=231 |       |
| X=88                  | x=120           | )=41                  |              | «=155  | ¶=182 | Ï=207 | è=232 |       |
| Y=89                  | y=121           |                       |              | «=156  | ·=183 | Ð=208 | é=233 |       |
| Z=90                  | z=122           |                       |              |        |       |       |       |       |

**C++:  
REVIEW  
FOR  
FINAL EXAM**

## C++ Notes

C developed in AT&T Bell Labs in 1972 for the UNIX operating System.

### Comments:

```
// are used for comments that go to the end of the line
/* Comments that don't */
```

### #define - defines constants and functions

```
(#define G 9.8)
#define abs(x) ((x) < 0) ? ((-x) : (x))
```

### #include -

Enables you to include the source lines from another file into the current one.

### Constant -

```
Const h = 2;
```

### Variables -

```
Int I; (integer)
Double x = 3.14 (fractions and decimals)
```

### Increment -

```
X++ //post-increment (after the fact)
++X //pre-increment (before the fact)
```

### Decrement—

```
x-- //post-decrement(after the fact)
--x //pre-decrement(before the fact)
```

ex:

```
int i, j;
k = 5;
```

```
k++ //k = 6 (k went up 1)
--k; //k = 5 (k went down 1 from 6 to 5)
```

```
k = 5;
I = 4 * k++; //k is 6 I = 20 (increment k before the operation)
```

```
K = 5
J = 4 * ++k; //k = 6 j = 24 (increment k after the operation)
```

Review Notes -- meanings

## Assignment Operations

|    |             |             |           |
|----|-------------|-------------|-----------|
| += | add to      | $x = x + y$ | $x += 12$ |
| -= | subtract to | $x = x - y$ | $x -= y$  |
| *= | multiply by | $x = x * y$ | $x *= 12$ |
| /= | divide by   | $x = x / y$ | $x /= 12$ |

## Input/Output of characters

Getche() – uses #include conio.h  
Gets a character from the keyboard and echoes the character on the screen

Getch() -- uses #include conio.h  
Gets a character from the keyboard but does not echo it to the screen

Getchar() -- #include stdio.h  
Gets a character from the keyboard and is portrayed in the header file

Puch() and putchar() -- #include conio.h  
Sends a character

## Screen – #include conio.h

Void clrscr(void); //clears the screen  
Void ccleol(void); //clears to the end of the line

## Cursor – #include conio.h

Void gotoxy(int x, int y); //moves the cursor location x, y  
Void wherex(void); //returns the column of x  
Void wherex(void); //returns the row of y

## Variables –

Char – character

Int -- integer

Float – single precision – 32 bits assigned for the number

Double – double precision – 64 bits assigned for the number

Char & Int can be described as unsigned or signed and long or short

The default for int is signed

The default for char is signed or unsigned

## Review Notes -- meanings

### Legal declarations –

Int i;

Char c;

Unsigned char uc;

Long int li;

Unsigned short int si;

Signed long si;

Variables declared outside the function are called GLOBAL

To make variables accessible to other files use EXTERN declaration

Local variables are declared as AUTO, STATIC or REGISTER

The default is AUTO

The auto variable is created when the program enters the function containing the auto declaration and is destroyed when the program exits the function

A static local variable is created the first time the function is called but continues and retains its value through subsequent returns and calls.

= assignment

\*= multiply by

/= divide by

% modulo by

+= add to

-= subtract from

&= and with

|= or with

<<= left shift by

>>= right shift by

() function call

[ ] array index

. member of a structure

-> member of a structure pointed to

? : ternary operator

, comma operator

ex: a ? b : c; If a is true the result of the operator is the value of b otherwise the result is the value of c.

## C++ Final Review Questions

**Directions:** Answer the following questions on the lines provided:

1. What is `stdlib.h` used for? \_\_\_\_\_
2. What is `iostream.h` used for? \_\_\_\_\_
3. What must start every program? \_\_\_\_\_
4. What will F7 do for your program? \_\_\_\_\_
5. What does `//` do for code? \_\_\_\_\_
6. What does `/* ..... */` do to code? \_\_\_\_\_
7. How would you use the tab in your program? \_\_\_\_\_
8. Instead of `endl`, how else can you end a line? \_\_\_\_\_
9. What does `int` mean? \_\_\_\_\_
10. What would you use `float` for? \_\_\_\_\_
11. What would you use “`iomanip.h`” for? \_\_\_\_\_
12. What is “`math.h`” used for? \_\_\_\_\_
13. What does “`%`” do? \_\_\_\_\_
14. What is “`conio.h`” used for? \_\_\_\_\_
15. What is “`ctype.h`” used for? \_\_\_\_\_
16. What would you use `char` for? \_\_\_\_\_
17. What is “`apstring.h`” used for? \_\_\_\_\_
18. What do you have to do in order to use “`apstring.h`”? \_\_\_\_\_

Name \_\_\_\_\_

Date \_\_\_\_\_

Period \_\_\_\_\_

**Test – C++**

Directions for 1 – 7: Match the header files with what they are used for.

- |                       |                                                          |
|-----------------------|----------------------------------------------------------|
| 1. _____ <iostream.h> | a. used for sqrt and pow.                                |
| 2. _____ <stdlib.h>   | b. used for random numbers                               |
| 3. _____ <math.h>     | c. used for toupper.                                     |
| 4. _____ <conio.h>    | d. for text input (letters or words).                    |
| 5. _____ <ctype.h>    | e. necessary for input and output.                       |
| 6. _____ "apstring.h" | f. for getch and clrscr.                                 |
| 7. _____ <iomanip.h>  | g. used for formatting (precision of decimals and setw.) |

Give the output to each segment of code for numbers 8 - 25:

8) cout<<"HELLO, I LOVE C++"<<endl;

8)

9) cout<<"It is March";  
cout<<" , Spring has started"<<endl;

9)

10) int num1=6;  
cout<<"num1 has the value..."<<num1<<endl;

10)

11) int num1=4;  
int num2=8;  
cout<<"num1<<" + "<<num2<<" is "<<num1+num2;

11)

12) int num1=20;  
int num2=14;  
cout<<"The product of num1 and num2 is..."<<num1\*num2;

12)

13) int num1=20;  
int num2=3;  
cout<<"The quotient of num1 and num2 is.."<<num1/num2<<endl;

13)

14) int num1=6;  
int num2=3;  
cout<<"The quotient of num1 and num2 is.."<<num2/num1<<endl;

14)

15) float num1=6;  
float num2=3;  
cout<<"The quotient of num1 and num2 is.."<<num2/num1<<endl;

15)

```
16) int num1=10;
 int num2=3;
 cout<<"The quotient of num1 and num2 is .."<<num2%num1<<endl;
```

16)

```
17) cout<<pow(3,4)<<endl;
```

17)

```
18) cout<<sqrt(9)<<endl;
```

18)

```
19) int a=4,b=8,c=9;
 if (a>b) cout<<"Hello"
 else cout<<"Goodbye";
```

19)

```
20) int a=4,b=8,c=9;
 if (a<b && c>b)
 cout<<"March"
 else cout<<"April"
```

20)

```
21) int a=4, b=8, c=9;
 if (a>c) cout<<"Monday"<<endl;
 else if (b<c) cout<<"Tuesday"<<endl;
 else cout<<"Wednesday"<<endl;
```

21)

```
22) for (int x=4; x<8; x++)
 cout<<x;
 cout<<endl;
```

22)

```
23) for (int x=0; x<=20; x+=2)
 cout<<x;
 cout<<endl;
```

23)

```
24) for (int x=15; x>=0; x-=3)
 cout<<x;
 endl;
```

24)

```
25) sum=0;
 for (int x=0; x<=5; x++)
 {
 sum+=x;
 }
 cout<<"sum is ..."<<sum<<endl;
```

25)

26) Write the code to ask for and read in an integer called num. The code should print out the value of num and whether num is positive or negative.

27) Write a loop to print the even numbers from 2 to 100

# C++ Review

CAN YOU DO THE FOLLOWING ON YOUR OWN?

Write a program that contains the following parts:

**Pause and clear the screen after each section**

1. Output your first name, skip 2 lines and output your last name
2. Input your first name, output it and skip 1 line
3. Write a function to input and a function to output your first name
4. Set up a for loop to output the numbers 1 to 10
5. Set up a for loop to output the numbers 2 to 20 even
6. Input 2 numbers and output their sum as a variable
7. Output 3 days and dates using the tab \t
8. Set up a while loop to enter letters. Q ends the loop
9. Input 1, 2 or 3 and use the case statement to check
10. Generate a random number between 1 and 50
11. Check to see if it is between 20 and 30
12. Check to see if it is less than 20 or greater than 30
13. Find the length of the word "computer"
14. Find the 3<sup>rd</sup> letter of the word
15. Find the 3<sup>rd</sup> and 4<sup>th</sup> letters of the word

## C++ REVIEW

1. Refer to prg 1. - THE PROGRAM HAS NO FUNCTIONS?

- a) TRUE
- b) FALSE

2. Refer to prg 1. - APSTRING IS NOT A RESERVED WORD.

- a) TRUE
- b) FALSE

3. Refer to prg 1. - THE USER IS PROMPTED TO ENTER HIS FULL NAME.

- a) TRUE
- b) FALSE

4. Refer to prg 1. - FIRSTNAME IS A C++ RESERVED WORD

- a) TRUE
- b) FALSE

5. COUT IS A C++ RESERVED WORD.

- a) TRUE
- b) FALSE

6. A PROGRAM STARTS WITH //

- a) ALWAYS
- b) SOMETIMES
- c) NEVER
- d) IT STARTS WITH \\  
\\

7. THE NAMES OF ALL FUNCTIONS BEGIN WITH A CAPITAL LETTER

- a) TRUE
- b) FALSE

8. EACH OPENING BRACE HAS A MATCHING CLOSING BRACE

- a) TRUE
- b) FALSE

9. THE WORD 'IF' CAN NOT BE USED AS A NAME FOR A VARIABLE

- a) TRUE
- b) FALSE

10. 'CHAR' IS NOT A RESERVED WORD

- a) TRUE
- b) FALSE

11. PROGRAM INPUT \_\_\_ COMES FROM CIN

- a) ALWAYS
- b) SOMETIMES
- c) NEVER

12. 7SEAS IS A VALID VARIABLE NAME

- a) TRUE
- b) FALSE

13. \_GET\_3 IS A VALID VARIABLE NAME

- a) TRUE
- b) FALSE

14. COUT IS A VALID VARIABLE NAME

- a) TRUE
- b) FALSE

15. XYZ IS A VALID VARIABLE NAME

- a) TRUE
- b) FALSE

16. EVALUATE THE FOLLOWING: COUT << 5 / 10 << ENDL;

- a) .5
- b) 5
- c) 50
- d) 500

17. EVALUATE THE FOLLOWING: COUT << 1 / 2 \* 10 << ENDL;

- a) .5
- b) 5
- c) 50
- d) 500

18. GIVEN: INT c = 3; COUT << c++ THEN c =

- a) 3
- b) 4
- c) 5
- d) 6

19. GIVEN a = 3, b = 4 AND INT c = a + b++ THEN c =

- a) 7
- b) 8
- c) 9
- d) 10

20. #INCLUDE <IOSTREAM.H>;

- a) IS CORRECT AS WRITTEN
- b) SHOULD BE: #INCLUDE <IOSTREAM.H>
- c) SHOULD BE: #INCLUDE <IOSTREAM>
- d) SHOULD BE: #INCLUDE <IOSTREAM.A>

21. COUT >> "HELLO WORLD" >>;

- a) IS CORRECT AS WRITTEN
- b) SHOULD BE: COUT >> "HELLO WORLD" >> ENDL;
- c) SHOULD BE: COUT << "HELLO WORLD" << ENDL;
- d) SHOULD BE: COUT << "HELLO WORLD" << ENDL

22. CIN << NUMBER;

- a) IS CORRECT AS IS
- b) SHOULD BE: CIN << NUMBER
- c) SHOULD BE: CIN >> NUMBER;
- d) SHOULD BE: CIN >> NUMBER

23. CIN << X << Y

- a) IS CORRECT AS IS
- b) SHOULD BE: CIN << X, Y <<;
- c) SHOULD BE: CIN X << Y ;
- d) SHOULD BE: CIN >> X >> Y ;

24. FOR (LOOP = 2; LOOP < 100; LOOP ++)

- a) IS CORRECT AS IS
- b) SHOULD BE: FOR (LOOP = 2; LOOP < 100; LOOP +1)
- c) SHOULD BE: FOR (LOOP = 2; LOOP < 100; LOOP +=1);
- d) SHOULD BE: FOR (LOOP = 2; LOOP < 100; LOOP =1)

25. WHILE (LOOP < 10);

- a) IS CORRECT AS IS
- b) SHOULD BE: WHILE (LOOP < 10)
- c) SHOULD BE: WHILE LOOP < 10;
- d) SHOULD BE: WHILE LOOP (<10);

26. X +=Y IS THE SAME AS

- a) X + Y
- b) X = X + Y
- c) X + Y
- d) NONE OF THE ABOVE

27. X +=1

- a) X = 1
- b) X + 1 = 2
- c) X = X + 1
- d) NONE OF THE ABOVE

28. FOR (LOOP = 2; LOOP < 10; LOOP +=2)

- a) WILL PRODUCE VALUES 2, 4, 6, 8, 10 FOR LOOP
- b) WILL PRODUCE VALUES 2, 4, 6, 8 FOR LOOP
- c) WILL PRODUCE VALUES 0, 2, 4, 6 FOR LOOP
- d) NONE OF THE ABOVE

29. FOR (LOOP = 2; LOOP < 10; LOOP +=3)

- a) WILL PRODUCE VALUES 2, 4, 6, 8, 10 FOR LOOP
- b) WILL PRODUCE VALUES 2, 5, 8 FOR LOOP
- c) WILL PRODUCE VALUES 2, 5, 8, 11 FOR LOOP
- d) NONE OF THE ABOVE

30.  $N = N + 1$  IS THE SAME AS

- a)  $N += 1$
- b)  $N + 1 = 0$
- c)  $N + 0 = 1$
- d) NONE OF THE ABOVE

31. `GETCH();`

- a) IS USED TO OUTPUT DATA
- b) IS USED TO INPUT A CHARACTER
- c) IS USED TO INPUT A WORD
- d) NONE OF THE ABOVE

32. `#INCLUDE <IOSTREAM.H>`

- a) IS USED FOR INPUT ONLY
- b) IS USED FOR OUTPUT ONLY
- c) IS USED FOR INPUT AND OUTPUT
- d) NONE OF THE ABOVE

33. `IF (X < 5) THEN COUT << "X IS LESS THAN 5";`

- a) IS CORRECT AS IS
- b) SHOULD BE: `IF (X < 5) THEN COUT >> "X IS LESS THAN 5";`
- c) SHOULD BE: `IF (X < 5) COUT << "X IS LESS THAN 5";`
- d) SHOULD BE: `IF (X < 5) COUT >> "X IS LESS THAN 5";`

34. `#INCLUDE <CONIO.H>` IS NECESSARY FOR

- a) `CIN`
- b) `CHAR`
- c) `CLRSCR()`
- d) `MAIN()`

35. `INT MAIN()` REQUIRES \_\_\_\_ AT THE END OF THE PROGRAM

- a) `END()`
- b) `END();`
- c) `RETURN 0`
- d) `RETURN 0;`

36. `IF (N > 5) THEN COUT << "ENTER ANOTHER NUMBER"`

- a) IS CORRECT AS IS
- b) SHOULD BE: `IF (N > 5) THEN COUT >> "ENTER ANOTHER NUMBER";`
- c) SHOULD BE: `IF (N > 5) COUT << "ENTER ANOTHER NUMBER";`
- d) SHOULD BE: `IF (N > 5) COUT >> "ENTER ANOTHER NUMBER";`

37. `INT` IS A TYPE OF VARIABLE

- a) TRUE
- b) FALSE

38. `DOUBLE` IS A TYPE OF VARIABLE

- a) TRUE
- b) FALSE

39. `BIG` IS A TYPE OF VARIABLE

- a) TRUE
- b) FALSE

40. `COUT << "X = " << X << ENDL;`

- a) IS CORRECT AS IS
- b) SHOULD BE: `COUT >> "X = " >> X >> ENDL;`
- c) SHOULD BE: `COUT << "X = " << X << ENDL`
- d) SHOULD BE: `COUT << "X = " << X <<`

41. TO SKIP A LINE DURING OUTPUT YOU CAN USE

- a) `ENDL;`
- b) `\N;`
- c) BOTH A AND B
- d) NEITHER A AND B

42. C++ RESERVED WORDS MAY BE USED AS VARIABLES

- a) ALWAYS
- b) SOMETIMES
- c) NEVER
- d) ONLY IN AN EMERGENCY

43. `WAGES = RATE * HOURS;`

- a) IS IN CORRECT C++ FORM
- b) SHOULD BE: `WAGES = RATE * HOURS`
- c) SHOULD BE: `WAGES += RATE * HOURS`
- d) SHOULD BE: `WAGES += RATE * HOURS;`

44. `INT NUM1, NUM2;`

- a) CREATES 2 VARIABLES THAT CAN STORE ONLY INTEGERS
- b) CREATES 2 VARIABLES THAT CAN STORE INTEGERS AND DECIMALS
- c) CREATES 2 VARIABLES THAT CAN STORE INTEGERS AND FRACTIONS
- d) CREATES 2 VARIABLES THAT CAN STORE ONLY DECIMALS

45. `(NUM1 != NUM2)` IS CODE FOR

- a) `NUM1 = NUM2`
- b) `NUM1 IS GRATER THAN NUM2`
- c) `NUM1 IS LESS THAN NUM2`
- d) `NUM1 DOES NOT EQUAL NUM2`

46. `COUT << "ENTER A NUMBER \N";`

- a) IS AN INPUT STATEMENT
- b) IS AN OUTPUT STATEMENT
- c) NEITHER A OR B
- d) BOTH A AND B

47. `FOR (I = 1; I < 10; I++)` IS REFERRED TO AS A

- a) BUNCH OF NUMBERS
- b) LOOP OF NUMBERS
- c) GROUP OF NUMBERS
- d) NONE OF THE ABOVE

48. WHILE AND FOR ARE BOTH

- a) RESERVED WORDS
- b) TYPES OF LOOPS
- c) BOTH A AND B
- d) NEITHER A AND B

49. \t IS USED

- a) TO TOTAL NUMBERS
- b) FOR TIME
- c) AS A TAB
- d) AN AN INPUT COMMAND

50. SETW IS USED

- a) TO CHANGE COLORS
- b) TO CREATE COLUMNS
- c) TO MOVE LINES
- d) TO SET UP A VARIABLE

51. IOSTREAM.H IS USED FOR

- a) INPUT
- b) OUTPUT
- c) BOTH A AND B
- d) NONE OF THE ABOVE

52. APSTRING.CPP IS USED FOR

- a) TEXT
- b) NUMBERS
- c) VARIABLES
- d) COMMANDS

53. TO SKIP A LINE USE

- a) ENDL
- b) /N
- c) NEW LINE
- d) NONE OF THE ABOVE

54. VOID IS USED FOR A(N)

- a) INCLUDE
- b) FUNCTION
- c) FILE
- d) NONE OF THE ABOVE

55. % IS USED FOR

- a) DIVISION
- b) REMAINDERS
- c) PERCENT
- d) DECIMALS

56. THE INT VARIABLE DECLARATION IS USED FOR

- a) ALL NUMBERS LESS THAN 32768
- b) INTEGERS LESS THAN 32768
- c) DECIMALS ONLY
- d) INTEGERS LESS THAN 1,000,000

57. THE LONG VARIABLE DECLARATION IS USED FOR

- a) DECIMALS ONLY
- b) FRACTIONS ONLY
- c) ALL NUMBERS
- d) INTEGERS ONLY

58. GIVEN: X = "ABCDEF": COUT << DATA[3] OUTPUTS ?

- a) 3
- b) ABC
- c) D
- d) EF

59. GIVEN: X = "ABCDEF": DATA.LENGTH() OUTPUTS ?

- a) ABCDEF
- b) DATA
- c) 6
- d) 4

60. GIVEN: DATA = "ABCDEF": DATA.SUBSTR(2,3) OUTPUTS ?

- a) 2, 3
- b) BC
- c) CDE
- d) CDEF

61. GIVEN: IF (X = 3 && Y = 2) && IS CODE FOR

- a) AND
- b) OR
- c) THEN
- d) NONE OF THE ABOVE

62. GIVEN: IF (X = 2 || Y = 3) || IS CODE FOR

- a) AND
- b) OR
- c) THEN
- d) NONE OF THE ABOVE

63. FOR ANSWERS TO BE ROUNDED TO THE NEAREST HUNDREDTH YOU

- a) SETPRECISION(1)
- b) SETPRECISION(2)
- c) SETPRECISION(.1)
- d) SETPRECISION(.01)

64. X IS 2 YEARS OLDER THAN Y. IF Y IS B YEARS OLD THEN X IS

- a) B
- b) B + 2
- c) B - 2
- d) 2

65. FIND THE VALUE OF X NICKELS, Y DIMES AND 3 QUARTERS

- a) \$1.25
- b) X + Y + 3
- c) 5X + 10Y + 75
- d) 5X + 10Y + 3

66. IF (X < 3);

- a) SHOULD BE: IF (X < 3) THEN
- b) SHOULD BE: IF (X < 3)
- c) SHOULD BE: IF (X < 3); THEN
- d) NONE OF THE ABOVE

67. C = 14, B = 10 AND A = 2: X = C - B/A THEN COUT << X;

- a) OUTPUTS 2
- b) OUTPUTS 9
- c) OUTPUTS 6
- d) OUTPUTS 3

C++ PART 2 Questions

|                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>#1</p> <p>When you are ready, load C++ and program the following:</p> <p>Generate random numbers for dimensions of a rectangle and have the user find the area and perimeter.</p> <p>The computer will grade the answer and give the correct answer if the answer is wrong. Add a way to do 5 problems.</p> <p>At the end of the program Your score will appear.</p> | <p>#2</p> <p>Generate 2 random numbers and have the user Find their sum.</p> <p>The computer will grade the answer and give the correct answer if the answer is wrong. Add a way to do another problem.</p> <p>Add a way to do 5 problems and Show your score when finished.</p>     |
| <p>#3</p> <p>Generate 2 random numbers and have the user Find their difference.</p> <p>The computer will grade the answer and give the correct answer if the answer is wrong. Add a way to do another problem.</p> <p>Add a way to do 5 problems and Show your score when finished.</p>                                                                                 | <p>#4</p> <p>Generate 2 random numbers and have the user Find their product.</p> <p>The computer will grade the answer and give the correct answer if the answer is wrong. Add a way to do another problem.</p> <p>Add a way to do 5 problems and Show your score when finished.</p> |
| <p>#5</p> <p>Create a cash register program Ask the user for an item, its cost and then figure out cost and sales tax (use 8.25%) Then ask the user for an amount and then calculate the change.</p> <p>Allow for another item using a WHILE. loop</p>                                                                                                                  | <p>#6</p> <p>Allow the user to enter a number The program will output whether the number is positive, negative or zero</p> <p>Allow for another number to be entered.</p> <p>Allow for another problem using A while loop.</p>                                                       |
| <p>#7</p> <p>Allow the user to enter 3 numbers. The program will output the sum of the first two times the third one.</p> <p>Allow for another set of numbers to be entered.</p>                                                                                                                                                                                        | <p>#8</p> <p>Allow the user to enter 1 number Add 4 to that number then multiply the result by 6. Subtract 7 from that result and then output the final result.</p> <p>Allow for another number to be entered.</p>                                                                   |
| <p>#9</p> <p>Allow the user to enter 1 number to start a loop and a number to end it. Output the loop and the sum of the numbers</p> <p>Allow for another set of numbers to be entered.</p>                                                                                                                                                                             |                                                                                                                                                                                                                                                                                      |

**ATTENDANCE**

**MAKE-UP**

**PROJECTS**

# Attendance Make-Up Projects:

## Directions:

- You must complete one project for every 2 absences over the allowed 28.
- **YOU MUST** see your teacher before beginning for more detailed instructions.

## Projects:

1. Winning Percentage of a Sports Team
2. Solving Equations in the Form  $ax + b = c$
3. Cash Register
4. Coin Flip
5. Addition Quiz
6. Grades
7. Miles Per Gallon
8. Rate of Pay
9. Equation of a Line
10. Solving Equations Menu Program

## Make-up #1

### Program to calculate the winning percentage of a sports team

You will input the number of wins and losses of a team, calculate the winning percentage and whether the team has a winning record or losing record.

**Input** - number of wins and losses

**Process** – calculate total number of games won and percentage  
 $\% = (\text{wins} / \text{total games}) * 100$

- make a decision on the percentage: > 50%, =50%, < 50%

**Output** – number of wins, losses, percentage and whether the team is a  
winning team, average team or losing team.

Plan:

Prompt for number of wins

Input number of wins

Prompt for number of losses

Input number of losses

Calculate total number of games

Calculate the percentage

Check to see if the percent is > 50%, = 50%, < 50%

Output the number of wins, losses, percentage and whether the team is a winner, average or loser.

Bonus: Enter the name of the team and have it appear with the record.

The results and team name should be the only data on the screen.

## Make-up #2

### Solving equations in the form of $AX + B = C$

You will be given a series of equations in the form of  $AX + B = C$  and have to solve each one for  $X$ . By solving this equation for  $X$  you will develop the algorithm for your program.

Input – values for  $A$ ,  $B$  and  $C$ .

Process – use the algorithm and solve for  $X$ .

Output – write the equation and the solution.

Plan:

Prompt for the value of  $A$   
Input the value of  $A$

Prompt for the value of  $B$   
Input the value of  $B$

Prompt for the value of  $C$   
Input the value of  $C$

Calculate the value of  $X$  (think about what values  $X$  could have)

Output – the equation and the solution

Try your program with the following equations:

- $2X + 6 = 13$        $X = 3.5$
- $5X - 9 = 21$        $X = 6$
- $12X + 34 = 47$        $X = 1.083$

Bonus: Add a second part to the program to solve equations in the form of  $AX + B = CX + D$

## Make-Up #3

# Cash Register

Write a program that will turn your computer into a cash register. Enter the number of items purchased and the cost of the item. Have the program calculate the tax (8.75%) and total due. Enter the amount given to the cashier, check to make sure it is enough and then calculate the change.

**Input** – number of items, cost of each item and cash tendered

**Process** – calculate the tax, total cost and change.

**Output** – the cost, tax, total and change

### Plan:

Prompt for the number of items  
Input the number of items

Prompt for the cost  
Input for the cost

Calculate the tax

Output the cost, tax and total

Prompt for the amount due  
Input the amount due

Check to make sure the amount is correct  
Calculate the change

Output the change

Ex:

Buy 15 widgets at \$2.45 = \$36.76  
Tax = \$3.22  
Total = \$39.98

Give the cashier \$40.00  
Change = \$.02

**\*\*ALSO: REPEAT THE PROGRAM UNTIL THE USER DECIDES TO QUIT!**

## Make-up #4

### Coin Flip

Write a program that will simulate flipping a coin 100 times. The user will enter the number of heads they expect and the program will output the actual number and how close they got.

Input – how many heads are expected

Process – make a decision whether heads or tails and calculate how close

Output - results

Plan:

Prompt for the number of heads

Input the number of heads

Set up a loop to generate a random number between 0 and 1 100 times. Zero will be heads and 1 will be tails. When the loop ends calculate how close the user was

Output the guess, the actual amount and how close the user was.

## Make-up #5

# Addition Quiz

Write a program that will generate 10 addition problems, allow the user to answer them and respond with a positive comment if correct or the correct answer if incorrect.

Directions – give the user an explanation of the program and directions.

Set up - Variables

---

Set up the random numbers and reset the score

Start the loop

Create numbers for the problem and a correct answer.

The user has to be able to read the question and respond with an answer.

The answer must be checked, score updated and a response output.

Pause before the next question.

End the loop

Output the score.

**BONUS (this will give you an extra day):** Set up a menu that will allow the user to do addition or subtraction and have the program repeat after a section has been completed. All scores must be reset. When the program ends, tell the user how many times he/she has run the program.

## Make-up #6

# Grades

Write a program which allows the user to enter a name and a set of grades. Assume the grades are out of 100 points each. The program will calculate the average and assign a letter grade to the score.

90 and over = A

80 – 89.9 = B

70 – 79.9 = C

65 – 69.9 = D

< 65 = F

Directions – Give the user an explanation of the program and directions.

Set up - Variables for

---

Input the name of the student

Set up loop – how many grades?

Input grades

Calculate sum

End the loop

Calculate the average

Find the letter grade

Output the data – Name, average, letter grade

Pause

Clear the data and output

End of the loop

## Make-up #7

# Miles per Gallon Project

You are going to write a program to calculate the miles per gallon a car gets during a trip as well as the data about the trip, gallons used and cost of the trip. During the trip you will be filling up the car 4 times at 4 different prices. Write functions to do each of the parts – get the mileage, get the fill-ups & calculate the cost, calculate the miles per gallon and output the results.

You will need:

1. The starting mileage
2. The ending mileage
3. The number of gallons at each of the 4 fuel stops
4. The cost of gas at each of the 4 fuel stops

Your program will calculate:

1. Total miles traveled
2. Total gallons used
3. Total cost of the trip
4. Miles per gallon

Variables needed: \_\_\_\_\_

Function 1 – Get the mileage

Function 2 – Get the gallons & price per gallon & calculate the trip cost

Function 3 – Calculate the miles per gallon

Function 4 – Output the results

Use the following data:

Starting mileage – 1523.6

Ending mileage – 3245.9

Fills and cost - #1 18.9 @ \$1.59 **(don't enter the \$ just the 1.59)**

#2 20.2 @ \$1.63

#3 15.6 @ \$1.61

#4 19.3 @ \$1.62

Results should be:

Total miles – 1722.3

Total gallons – 74

Total cost - \$119.359 or \$119.36

Miles per gallon – 23.2743

## Make-up #8

# Rate of Pay Project

Write a program that will calculate the gross pay, taxes and net pay of an individual under the following conditions:

1. An fixed hourly wage for working 40 hours a week or less
2. 1.5 times the hourly wage for hours over 40 up through 80
3. 2 times the hourly wage over 80 hours
4. Federal tax of 28%
5. State tax of 8%
6. F.I.C.A. tax of 7.65%

- Plan:
1. Input the person's name
  2. Input the rate per hour
  3. Input the number of hours worked
  4. Calculate the gross pay
  5. Calculate the taxes
  6. Calculate the net pay
  7. Output the data neatly
  8. Bonus – get the program to go over again a fixed number of times

Try your program with this data:

- Mr. A. - 35 hours at \$6.50 per hour
- Mr. B. - 45 hours at \$8.50 per hour
- Ms. C. - 90 hours at \$10.00 per hour

Output:

Mr. A 35 hours @ \$6.50  
Gross pay - \$227.5  
Federal tax - \$63.70  
State tax - \$18.20  
F.I.C.A. tax - \$17.40  
Net pay - \$128.20

Mr. B 45 hours @ \$8.50  
Gross pay -  $\$340 + \$63.70 = \$403.75$   
Federal tax - \$113.05  
State tax - \$32.30  
F.I.C.A. tax - \$30.89  
Net pay - \$227.51

Ms. C 90 hours @ \$10.00  
Gross pay -  $\$400.00 + \$600.00 + \$200.00 = \$1200.00$   
Federal tax - \$336.00  
State tax - \$96.00  
F.I.C.A. tax - \$91.80  
Net pay - \$676.20

## Make-Up #9

# Equation of a Line

Write a program to find the equation of a line given 2 points:

Use:  $Y = MX + B$  - M is the slope, B is the Y intercept

Plan: 1. Input 2 points – enter x then y then the other x and other y  
2. Find M [  $(y_2 - y_1) / (x_2 - x_1)$  ]  
3. Find B – substitute M, x1 and y1 into  $Y = MX + B$  and solve for B  
4. Output the equation

Consider: What if  $(x_2 - x_1) = 0$  or  $M = 0$

Try your program with the following data:

1. (1, 5) and (3, 11) – you should get  $Y = 3X + 2$
2. (2, 5) and (2, 9) – you should get  $X = 2$
3. (2, 4) and (5, 4) – you should get  $Y = 4$

Bonus: Get the program to repeat until the user decides to quit.

## Make-up #10

# Solving Equations Menu Program

Earlier you developed a program to solve equations in the form  $ax + b = c$ . You will now expand that program into a menu program where the user can choose to solve equations in the following forms:

$$\begin{aligned}x + a &= b \\x - a &= b \\ax + b &= c \\ax - b &= c \\ax + c &= cx + d\end{aligned}$$

The user will be presented with a menu and upon choosing a type of equation, will be sent to a section of the program that does that type. They will also be given a choice to quit the program. Each section should be done using functions.

All answers will be rounded to the nearest hundredth.

Steps:

1. includes – don't forget rounding
2. define the functions
3. start the main program
4. declare the variables
5. set up the conditions for the loop
6. output the equation choices
7. accept the choice
8. go to the function, input the data and output the result
9. end the loop
10. end the main program
11. start the functions
  - a) get the data
  - b) process the data
  - c) output the result

# C++ Final Project

You will write a program that will serve as a tutorial for the beginning C++ programmer. Your program will describe a term and show how it is used by giving an example:

Ex:

Line 1 – cout is used to put text and/or data on the screen

Line 2 – cout<< “Text goes inside of the quotes – data outside”<<endl;

Above is what appears on the screen when you run your program.

Below is how you write the code:

Code for line 1 → cout<< “cout is used to put text and/ or data on the screen”<<endl;

Code for line 2 → cout<< “cout<<\” Text goes inside of the quotes – data outside \” ”<<endl;

Terms to be included:

\_\_\_\_\_ Includes: iostream, iomanip, conio, apstring, fstream

\_\_\_\_\_ variables: int, double, apstring, float

\_\_\_\_\_ cout – text only, \n, \t

\_\_\_\_\_ cout – text and variable

\_\_\_\_\_ cin

\_\_\_\_\_ if – then

\_\_\_\_\_ for loop

\_\_\_\_\_ while loop

\_\_\_\_\_ instream

\_\_\_\_\_ ostream

\_\_\_\_\_ clear screen

\_\_\_\_\_ getch

\_\_\_\_\_ setw

\_\_\_\_\_ .find

\_\_\_\_\_ .fail

\_\_\_\_\_ .substr

\_\_\_\_\_ random numbers

\_\_\_\_\_ increment a variable

\_\_\_\_\_ decrement a variable

\_\_\_\_\_ function

\_\_\_\_\_ case

**Backslash + “  
-- prints a “ on the screen**

YOUR PROGRAM MUST RUN WITHOUT ERRORS AND COVER EACH OF THE TERMS LISTED ABOVE – ORDER IS NOT IMPORTANT.

THIS WILL SERVE AS REVIEW FOR YOUR EXAM.

AFTER THE EXAM YOU WILL DELETE FILES → ALL FILES THAT DON'T END IN CPP OR IDE IN YOUR HOME DIRECTORY. YOU WILL KEEP CPP AND IDE FILES FOR HELP WITH YOUR OTHER PROJECTS.