MINI 1 SAMPLES that all earned a 5/5 evaluation.

NOTES:

a. All these samples happen to be from the same MINI 1 submission of a single student.
b. This student used HOOVER as required.
c. This student preferred to write a précis for each source. That is one method. REMEMBER, It is also possible and acceptable to write a single, longer précis, covering all the sources for one topic.

1. Data types in C. How many? What are they? Features of each?

Hoover précis – There are four basic datatypes in C – int, float, double and char. The int, double and float datatypes are intended for numeric values. The precision possible is different depending on the data type with the smallest being the int(intended to store whole real numbers), the next largest precision comes with the float data type(intended to store a large range of real number) and with the largest precision being the double(which is intended to store the largest range of real numbers, it contains roughly twice the precision of floats according to Hoover). The char datatype is used to store character symbols and the metadata required to display text.

http://en.wikipedia.org/wiki/C_data_types – The C language contains many basic data types. Most of these data types are from one of the four basic arithmetic type specifiers in C(int, float, double, char) and optional specifiers. The int, float and double data types are intended to store real number values in different precisions. The optional specifiers extend or limit these types. For example and unsigned int can contain a larger positive value than a signed int, but cannot contain an negative number. The long and short modifiers define different amounts of storage allocated to the variable. A short int will be allocated less than or equal amount of storage as an int which will be allocated less than or equal amounts of storage as a long int. The actual size of int datatypes varies by implementation, the standard only specifies the size relation between data types and the minimum sizes for each data type.

http://www.lix.polytechnique.fr/~liberti/public/computing/prog/c/C/CONCEPT/data_types.html – Datatypes are used to define and allocate memory to a variable before it's use. There are 6 basic data types in C – int, float, double, char, void, enum. C does not natively have a Boolean data type. Each data type is used to define different kinds of data/values. The int data type, which should be the smallest most efficient data type, is used to define integer numbers. The float and the double are used to define floating point numbers with the double being intended for a larger range of floating values than the float data type. The char is used to store character symbols and information. The void and enum data types appear to be largely related to functions. The void keywork allows the programmer to create functions that either do not take or do not return a value. The enum keyword/data type allows the programmer to define a list of aliases which represent integer numbers, this seems to behave similar to a hash or hash table in other languages.

Summary – There are 4 basic datatypes used in C that are extended to limited(or more closely managed) by using specifiers. These base data type can be split into two groups – the ones intended to store numeric values and those intended to store character symbols and text information. The numeric data types include int(used to store whole, real numbers), float(used to store floating point numbers) and double(used to store large floating point numbers. The data type used to store character symbols and text information is the char data type. In defining a variable with these data types the programmer is defining not only the variable but also asking C to allocate a specific size of memory for each data type. The use of specifiers allows this memory allocation to be more closely managed. The void and enum keywords may also be considered basic data types by some.

2. Conditionals in C. What are they? What do they do? When/Why should we use them?

Hoover précis - The basic conditional in C is the if-else block. In the if condition programmers can use ==(equal), !=(inequal), <, >, <= and >=. Mutiple conditional statements can be chained together using the and(&&) and the or(||) operators. Statements in the code are enclosed within {}s, if brackets are not used the conditional applies only to a single following statement.

http://www.cs.cf.ac.uk/Dave/C/node5.html - Conditionals are methods that allow the programmer to control the flow of logic in a program. For the most part these are very similar to conditionals in other languages. There are 5 logical operators in C: ==, !=, ||, &&, and the unitary operation not !. These logical operators are used in conjunction with if, if-else, else-if and other similar conditional operations. The ternary condition(?) can also be used to express simple if statements. It is used in the form:

expression ? true_result : false_result

Programmers can also use the switch statement to allow multiple choice of selection items at one level of a conditional. This is visually far neater than writing multiple level if-else statements.

http://en.wikibooks.org/wiki/C_Programming/Control - Conditionals allow the computer to demonstrate basic decision-making skills which allows for more meaningful and complex programs. Conditionals require the use of relational and equivalence expressions. These include: <, >, <=, >=, ==, and !=. As C does not contain a Boolean it is common to define TRUE as 1 and FALSE as 0 to allow the use of the relational and equivalence expressions. In addition to the relational and equivalence expressions, logical expressions(&&, ||, and !) are used to chain conditional statements together into larger, more complex statements. C uses short circuit evaluation of logical expressions, it evaluates chained conditional statements in order and can be used to prevent array index errors and similar if used intelligently. There are two primary conditional statements used in C – the if-else statement and the Switch-Case statement. The switch-case statement can be considered in terms of nested ifs and allows for greater clarity in reading the code.

Summary – there are two types of conditionals. The IF-ELSE statement which can be nested in a variety of ways to increase the complexity of the code flow and the SWITCH-CASE statement which allows the programmer to allow a multiple choice of selection items at one level of a conditional. The SWITCH-CASE statement could be built manually using the IF-ELSE statement and is mostly useful in reader clarity. In order to provide the logic that triggers the conditional it is necessary to use relational and equivalence expressions(which include <, >, <=, >=, ==, and !=). Relational and equivalence expressions can be chained together into larger logical statements using logical expressions(which include &&, || and !). The use of conditionals allows programmers to develop much more complex computer programs.

3. Loops in C. What are they? What do they do? When/Why should we use them?

Hoover précis - There are three types of loops in C, the for, the while and the do-while. The for loop is intended to be used to implement a loop such that it executes a fixed or predefined number of times. In order to accommodate this it is given both an initialization conditional and an ending conditional. The do and do-while loop are intended to be used to implement a loop such that it executes an unknown number of iterations, to accommodate this do and do-while loops are only given an ending conditional. The difference between a do and a do-while loop is that the do loop may, if it fails the conditional on the first attempt, not execute the loop whereas the do-while will always execute the loop at least once.

http://www.cprogramming.com/tutorial/c/lesson3.html - Loops are used to repeat a block of code. Using loops allows the programmer to write a very simple statement to produce a significantly greater result simply by repetition. For loops are the most basic type of loops. The syntax is for(variable initialization; conditional; variable update){} The variable initialization allows you to either declare a variable and give it a value or give a value to an already existing variable The conditional tell the program that while the conditional statement is true the loop should continue to repeat itself. The variable update is the easiest way for the loop to handle changing or updating the loop variable. Each of these sections may be empty(though the semi-colons must be present) and if the conditional section is empty the loop will evaluate to true and will run until something else stops it. Do-While loops are useful for things that the programmer want to loop at least once. In a do-while the conditional is tested at the end of the block instead of the beginning so the block will execute before the conditional is evaluated. A while loop says 'Loop while the condition is true and execute this block of code.' A do-while loop says 'Execute this block of code and then continue to loop while the conditional is true.' Two important keywords to looping are the break and continue keywords. The break keyword will exit the most immediately surrounding loop regardless of what the conditions of the loop are. The continue keyword will stop its current iteration, update its variables/itself, and continue from the beginning of the loop.

http://www.tutorialspoint.com/cprogramming/c_loops.htm - Loops are a control structure that allow for a more complication execution path. They allow the programmer to execute a statement or group of statements multiple times. There are four loop types. The while loop repeats a statement while a given condition is true(it evaluates the condition before executing the body). The for loop executes a statement or group of statement multiple times and abbreviates the code that manages the loop initialization, loop conditional and iteration. The do-while loop is similar to a while loop except it tests the conditional at the end of the loop body. Nested loops – all of the other loop types can be nested to create more complex code control flows. In addition to the four loop types there are also three loop control statements. The Break keyword terminates the loop(or switch) statements and moves to the statement immediately following the loop. The Continue keyword causes the loop to skip the remaining body of the loop and immediately reset its conditions and continue iterating from the beginning of the loop/loop body. The goto statement transfers control to the labeled statement, this is not recommended for use.

Summary – There are three major types of loops and these loops can be nested together in a variety of ways to result in a more complex program. The for loop should be used when the code must iterate a specific number of times. It can use external variables or programmers can use the structure of the loop to define the looping variable, it's iteration behavior and it's conditional. Any of the statements in the loop control can be left out but this can result in unexpected behavior if not used carefully. Not including a conditional in the loop control will result in a loop that executes infinitely until it is stopped by something else. The while loop tests the given conditional and then executes the loop body. While loops may not execute any code in the body if the conditional evaluates to false in its first run. The Do-While loop is similar to the while loop but guarantees that the loop body will be run at least once. It runs the loop body and then evaluates the conditional. In addition to the different loop types there are two keywords that are very valuable to controlling the code flow in loops. The break keyword stops the current iteration and exits the most containing loop. The continue keyword causes the loop to skip the rest of the loop body, update and reevaluate the conditional/iteration and continue the loop from the top of the loop body.