Basic C Programming 1.

1. What is a compound statement? How is it set off in a C program?
    a. Compound statement are statements that are bounded inside a { } bracket. For example, if/while/do-while statements. If the condition is met, it will be set off.
2. What is the use of comments in C?
    a. To enable either the users or someone else to understand the code.
3. What is a NULL statement?
    a. A statement that contains only a " ; "
4. What is the comma operator?
    a. Acts as a separator
5. What is the modulus operator?
    a. Takes the modulus of a number.
        i. E.g 10modulus3 is 1
6. State what is meant by mixing data types?
    a. Using different data types with another different data type
        i. E.g Dividing an int with a double
7. What happens if a type int is added to a type float?
    a. The int value will be changed to a float and then added to the float type data
8. What are the rules for mixed data types in C?
    a. The operand with lower rank will be converted to the operand with a higher rank
    b. https://overiq.com/c-programming-101/implicit-type-conversion-in-c/
9. What is meant by a cast in C?
    a. Converting a data type into another data type
        i. E.g mean = **(double)sum** / count;
10. What is a preprocessor directive?
    a. Is a macro processor that are lines included in a program that begins with # (E.G #include or #define) They are invoked by the compiler to process some programs before compilation
11. Explain what is meant by the #include directive.
    a. Is a way to include a standard or user-defined file in the program
12. State the purpose of the printf() function.
    a. Print data on to the output screen
13. How are characters distinguished from strings?
    a. Characters use single quotation marks such as 'C'
    String uses double quotation marks such as "C"
14. What is the purpose of a format specifier in the printf() function?
    a. To define the type of data to be printed
15. What is an argument?
    a. Is the actual value which is passed to the function
        i. E.G sum(5,4) ---> 5 & 4 are the arguments

16. State the purpose of the scanf() function.
    - Read in a value key in by the user that ends with a new line “\n”.

17. How does the scanf() function know what variable identifier to use for inputting the user data?
    - The first argument in scanf() function specifies the type of variable that is expected from the user (eg: scanf(“%d”,&num1); %d represents a decimal integer)

18. Explain the use of %i format specifier w.r.t. the scanf() function?
    a. %i is able to take in values in hexadecimal or octal
19. What is typecasting?
    a. Converting one datatype into another
20. What is the difference between declaring a header file with < > and “ ”?
    a. “ “ is used to define user-defined header files
    b. <> is used to define system header files

Branching

1. What are relational operators?
    a. They are used to compare values of two expressions
2. State all of the relational operators used in C.
    a. < Less than
    b. > Greater than
    c. <= Less than or equal to
    d. >= Greater than or equal to
    e. == Equal to
    f. != Not equal to
3. What are the two conditions that relational operators may have?
    a. Must be both arithmetics or
    b. Must be both strings
       0 or 1

4. Explain what is meant by the statement that a relational operator in C returns a value.
    a. Returns either a 0(False) or 1(True)
5. Explain the operation of the if statement.
    a. The condition will be tested first. If it’s true, the statement in the compound statement will be executed.
6. Describe the action of if-else if-else statement in C.
    a. The condition will be tested first. If it’s true the if’s compound will be executed first. If the test is failed, the else’s compound will be executed.
7. State the purpose of the switch statement in C?

    a. Compares the value with multiple cases. Once the case match is found, the block of statements associated with that particular case will be executed

8. What other keywords must be used with the switch statement?
    a. Switch, case, default(optional), break (optional)

9. State the purpose of the keyword default in the switch statement?
    a. Should any of the cases not be matched with the value, the default block will be executed
10. State the effect of omitting the break in a C switch.
    a. If the break is omitted, the program will execute all the cases that are found after the matched case
11. What do you need to do in order to make sure that default is always executed in a C switch?
    a. Write the default block at the beginning of the switch clause.
12. Explain the operation of the conditional operator.
    a. `expression1 ? expression2 : expression3`
        i. `If expression 1 is true, expression 2 will be executed`
        ii. `If expression 1 is false, expression 3 will be executed`
13. What values make the conditional operator TRUE? What values make it FALSE?
    a. True = 1 or any non zero value.
    b. False = 0
14. Does a break is required by default case in switch statement?
    a. Not required but should always have
15. When is a "switch" statement preferable over an "if" statement?
    a. If there are multiple alternatives to be selected
16. State the difference between the C operation symbols = and ==.
    a. = is an assignment
    b. == operates on two operands

Looping

1. What are the major parts of a C for loop?
    a. Initialization (x=0)
    b. Condition (x<size)
    c. Change (x++)
2. Can you have more than one statement in a C for loop? Explain.
    a. Yes, if there are more than one statement, { } are required
3. Explain the meaning of ++Y.
    a. Increment Y **FIRST** then do the operation
4. State the construction of the C while loop.
    a. The condition has to be set-up first
    b. Then statements to be run if the condition is true
5. Under what condition will a while loop be repeated?
    a. When the condition is true
6. What is the loop condition tested in a C while loop?

    a. True(1) or False(0)
7. What is a good use of a while loop?
    a. When you don’t know how many times a loop will actually execute
8. State the construction of the C do-while loop.
    a. First, write the statements that will be run initially
    b. Then create the condition
    c. Should the condition not be satisfied, the loop will end
    d. Repeat Step C until satisfy
9. Under what condition will the do-while loop be repeated?
    a. When the condition is True
10. When is the loop condition tested in a C do while loop?
    a. The loop condition will be tested after the do has been executed
11. What is a nested loop?
    a. A loop that is within another loop
12. Explain the structure that should be used with nested loops?
    a. For loops within a For loop
13. Which of the three C loop types may be nested?
    a. For Loop, while loop, do-while loop
14. Which is preferred, the do-while or the while loop? Explain.
    a. While loop checks for the condition first
    b. Do-while loop will run the iteration statements first then check the condition
15. State a potential problem that accompanies use of the C do loop.
    a. It will execute the statement at least once before checking the condition

Functions

1. Name two ways of passing a value from a called function to the calling function.
    a. Call by value or call by references(Pointers)
2. What is the difference between a function prototype and a function definition?
    a. A prototype omits the function body while the definition has the function body
3. What is the limitation of the C return() used in a called function?
    a. It is only able to return 1 value
4. State how more than one value can be returned to the calling function from the called function.
    a. By using pointers or called by references
5. State why it is necessary to use separate functions in a C program?
    a. Reusability
    b. Divides a code into modules
6. What is the name of a variable declared within a function whose life lasts only as long as the function is active?
    a. Local variable
7. What is meant by a local variable?
    a. Are variables that are only available inside the function in which they are defined in. The variables are destroyed the moment the function ends.

8. State the meaning of the term “scope” as applied to variables in C.
    a. Scope refers to the accessibility of a variable in a given code. There are 2 types of scope variable, Local & Global.
9. What is the life of an automatic variable for the duration of the program?
    a. Automatic variables are local variables whose lifetime ends when execution leaves their scope and are recreated when the scope is reentered.
    b. E.g When it leaves for **X** function, the automatic variable is destroyed.
    c. It will be recreated when the **X** function ends
10. What is the name of a variable that is declared at the beginning of the program before main()? What is the scope of such a variable?
    a. Global variable
    b. They can be accessed anytime and anywhere
11. How is a global variable declared? A local variable?
    a. Global variable are declared before the main body/function
    b. Local variable are declared within the function
12. Is it considered good programming practice to use global variables? Explain.
    a. No. For large sized programs, that makes it hard to encapsulate everything.
13. If global variables are not used, state how values may be passed between functions.
    a. Pointers or call by value
14. Explain local static variables and their use?
    a. Local static variables can only be accessed inside their scopes only.
    b. Life of a static variables only ends when the program end.
    c. It will not be created/destroyed when another function is called
15. What is the difference between an actual parameter and a formal parameter?
    a. Actual parameters are values/variables passed while calling a function(Actual Values)
    b. Formal parameters are variables written/declared in function definition/prototype and receive their values when a call to that function is made
16. What is the difference between a local variable and a global variable?
    a. Local variable can only be accessed in the function it is being declared on
    b. Global variables are declared before the main function and can be used anytime and anywhere
17. Explain modular programming
    a. Is a technique that separates the functionality of a program into independent and interchangeable modules. E.g Functions are modules
    b. Helps in the ease of use, reusable and ease of maintenance

Pointers

1. What is address operator in C? How is it used?
    a. & is the address operator
    b. It is used to determine the location or the address of a variable
2. What is the indirection operator in C? How is it used?
    a. *

    b. Int x; int *p = &x; *p=5; then x = 5
3. What is a pointer?
    a. A pointer is a variable whose value is the address of another variable
4. Why a pointer is called a pointer?
    a. Cause it points to the address of another variable
5. How does a pointer get the address of another memory location? Give an example.
    a. By declaring a pointer variable then assigning the variable with the address of another variable
    b. Int x; **int *p = &x**; *p=5; then x = 5
6. How is a pointer declared?
    a. Int *p
7. Explain the use of the & operator in returning values in the arguments of a called function.
    a. By calling a function with the & operator, the function will “create” pointers that are pointing to the variable’s address that are declared with the &
8. What must be in the formal argument of a function definition that is to return values to the calling function through its argument?
    a. “*”
9. Describe the mechanism for passing values to the called function that uses pointers in its formal argument.
    a. Pass by reference is a method in which rather than passing direct value, the address of the variable is passed as an argument instead to the called function
10. What terminology is used when an address operator is used to assign values to function arguments?
    a. Pass by reference
11. State how more than one variable is passed from a called function to the calling function?
    a. By using Pass by reference
12. What is a NULL pointer?
    a. A pointer that does not point to any object or function
13. What is a pointer on pointer?
    a. A pointer is being pointed to a pointer which is pointed to the address of a variable
14. How do you pass a value to a variable by using a pointer? What must you make sure of before doing this? Give an example?
    a. Int x; **int *p = &x**; *p=5; then x = 5
    b. Make sure the pointer is created with a * sign
    c. Make sure the pointer is assigned to a variable’s **address**
15. What is the difference between call by value and call by reference?
    a.

Call by value	Call by reference

Copies the value to the function	Copies the address of a variable to the function
Modified value are stored in different memory locations	Modified value will be stored at the same address
Original value is not modified	Original value is modified
A copy of the variable is passed to the function	The variable itself is passed

Arrays

1. What is an array?
    a. A variable that can store multiple of the same type of data in contiguous memory location
2. State how you would declare a numeric array of ten elements in a C program?
    a. Int ar[10];
3. What is the index of the first element of the array?
    a. 0
4. What is the range of array integer index?
    a. 0 to n-1
5. What is the advantage of array?
    a. Collection of similar types of data
    b. Only have to remember the first index of array
    c. 2D array to represent a matrix
    d. E.g If you want to store the marks of all students, it will be easier to store in an array
6. What is the disadvantage of array?
    a. Wastage of memory because arrays are fixed in size
    b. Not possible to increase the size of array once declared
7. Explain what programming method was used in order to get arrayed values from the program user.
    a. `for (x=0; x<4;x++)`
    b. `    {`
    c. `        printf("Enter number %d \n", (x+1));`
    d. `        scanf("%d", &num[x]);`
    e. `    }`
    `f. X being the number of arrays`
8. What is an easy method of getting user input values into an array?
    a. Using for loops
9. What is the meaning of base address of the array?
    a. Address of the first element of an array is called the base address
10. Can a pointer access the array?

    a. Yes, the pointer will point to the base address and then + any number to get to the address the user need
11. How are arrays passed to functions?
    a. Either by call by value or reference
    b. To pass an entire array, you can do so by passing the address of the first array and then the size of the array
        i. E.g myfunction(var_arr,7);
        ii. Var_arr is the array name and 7 is the size of the array
12. If an array passed as an argument to a function, what actually gets passed?
    a. The base address
13. Is the name of an array a pointer variable or a pointer constant?
    a. Pointer constant
14. How many dimensions need to be specified when passing a multi-dimensional array as argument to a function?
    a. The first array dimension does not have to be specified but the second dimension must be given
15. State the method used to cause a series of entered values to be displayed in the opposite order from which they were entered.
    a. Use a for loop
    b. for(x= arraysize-1; x >= 0; x--)
    c. Then printf array[x]
16. What method was used in order to extract a minimum value from a list of entered values?
    a. For loop then if loop
    b. AKA linear search
17. How can the maximum value be extracted from a list of values?
    a. For loop then if loop
    b. AKA linear search
18. Explain the difference between a pointer variable and a pointer constant.
    a. Constant Pointers is a pointer that cannot change the address its holding
        i. E.g int * const ptr;
    b. A pointer variable can change the address to any variable
19. What is meant by row-major order in regard to the storage of matrix elements?
    a. Row1 is stored first followed by row 2 then row 3 and so on
    b. E.g ar[0][1] -> ar[0][2] -> ar[0][3] -> ar[1][0] and so on

Character Strings

1. State what is meant by a string?
    a. An array of characters
2. Explain how you indicate a char string in C.
    a. Char str[]
3. What is the use of ‘\0’?
    a. To let the compiler know that it is the end of the string

4. For a string consisting of five characters, how many array elements are required? Explain.
    a. 6. Another character is need for ‘\0’
5. What is the element number of the first character in a C string array?
    a. 0
6. Explain the relationship between pointers and string array elements.
    a. The pointer will point to the first character of the string
7. What is the difference between “\0” and ‘\0’?
    a. “\0” is an empty string
    b. ‘\0’ is a single character 0
8. Why use the statement char string[80]=”Hello” instead of char string[]=”Hello”?
    a. By using the 2nd one, the size of string will be 6 and it will not be possible to change the size of the string
9. Why is it not necessary to specify the string size when passing a string to a function?
    a. It is because the base address will be passed on the function and the function will know that it reached the end of the string when the address has the variable ‘\0’
    b. We can use the function strlen() to get the size of the string. (need to #include <string.h>)
10. What is actually passed between functions when one is dealing with strings?
    a. Base address of the string
11. (*)Where must storage space for a string be reserved when strings are passed between functions?
    a.
12. What would happen when processing a string which was missing a terminating null character?
    a. The compiler will keep searching the memory until it finds a null character(‘/0’) or hits an address that causes a memory protection fault
13. What are the differences between scanf() and gets() in relation to string input?
    a. Scanf
        i. will read input until it encounters a whitespace or new line
        ii. Used to read input of any datatype
    b. Gets
        i. will read input until it encounters newline, it consider white space as a character
        ii. Only for string input
14. List the common string operations?
    a. strlen() - Get the len of a string
    b. strcpy() - Copy a string
    c. strcat() - Combines 2 strings together
    d. strcmp() - compare 2 strings
15. If the null character is missing from the end of a character string, what does strlen() do?
    a. It will keep counting in the memory until it reaches ‘\0’
16. What happens to each string used in a strcat() operation?

    a. strcat(string1,string2)
    b. String1 will be the destination and will be edited
    c. String2 is the source and will not be edited

17. What is meant by C character classifications in ? List the common functions that check alphanumeric characters.
    a. Are operation to test characters for membership in a particular class of characters such as alphabets, control
    b. Part b
        i. isspace - checks for whitespace
        ii. isalpha - checks for alphabets
        iii. isdigit - checks for decimal digits
18. What is the function toupper() do?
    a. Converts a lowercase alphabet to a uppercase alphabet
19. How do you convert strings to numbers in C?
    a. sscanf(str,%d,&x)
        i. Str is the string that you want to convert to
        ii. %d - integer
        iii. &x is the variable that you want to store the result
20. What is the purpose of the function sprint()?
    a. Saves formatted output into a buffer
    b. `sprintf(buffer,format,variables);`
21. What is a rectangular array of characters? How is it stored in memory?
    a. An array of characters stored in N rows by P columns
    b. One row followed by another row
22. What is a ragged array?
    a. An array of arrays
23. What are the strength and weaknesses of rectangular and ragged arrays?
    a. Array of arrays (jagged arrays) are faster than multi-dimensional arrays and can be used more effectively. Multidimensional arrays have nicer syntax.
    b. Multi dimensional arrays --> array[x][y]
24. What is the difference between islower() and tolower() function in C?
    a. islower() checks whether a alphabet is in lowercase and returns a 1 if it's true and a 0 if it's false
    b. tolower() converts a uppercase letter to a lowercase letter
25. What is the difference between 'G' and "G"?
    a. 'G' means a character byte containing G
    b. "G" means a character byte which holds an G and a NULL '\0' character

Structures

1. Define a C structure.
    a. Is an aggregate of values. Their components are distinct and may possibly have different data types, including array and other structures

2. (*)Explain what the C member of operator does. What symbol is used for this operator for a simple structure?
    a. It is used to access a member of the structure
    b. "." or Dot
3. Describe what is meant by a structure tag.
    a. The structure tag is the identifier/keyword of the structure
    b. Is the word after the keyword struct
4. Can a structure tag be a variable type? Explain.
    a. No, the tag is a identifier for the specific structure as such words reserved by the variable type, i.e. int, char, cannot be used.
5. What is the purpose of the -> symbol in C as applied to C structures?
    a. It is to access the member of the structure via pointers.
6. How is a structure member pointed to when a structure pointer is used?
    a. https://overiq.com/c-programming-101/pointer-to-a-structure-in-c/
    b. `// declaring a pointer to a structure of type struct dog`
    c. `struct dog *ptr_dog`
    d. `struct dog spike; // spike is the name of the structure and dog is the tag`
    e. `ptr_dog = &spike;`
    f. `Ptr_dog now points to the structure variable spike`
    g. `Now, spike can be accessed using ptr_dog`
7. State the three operations that are allowed with structures.
    a. Structure pointer operator: ->
    b. Indirection operator: *
    c. Member of operator: .
8. What is a nested structure?
    a. A structure within a structure
9. Can a structure be an array? Explain.
    a. Yes.
    b. Declaring a structure : **struct personTag**
       Char name[40],id[20],tel[20]
    c. Then to declare a array structure, use **struct personTag student[10]**
    d. Array index is used to access individual elements
    e. To access a member of the specific element, use student[i].name
10. (*)Can a C structure have a member that is another structure? Explain.
    a. Yes it is possible.
    b. By using nested structure
11. (*)Can a C array of structures contain an array as one of its members? Explain.
    a. Yes, nested array in structure
12. (*)Can a C array of structures be a member of another array of structures? Explain.
    a. Yes
13. Does the C typedef create a new data type? Explain.
    a. No, it just give a type a new name

    b. See 14

14. State the purpose of the C typedef.
    a. Enhance program documentation
    b. Makes program easier to read and understand
    c. Define simpler data types for complex declarations such as structures
    d. E.g : `typedef unsigned char BYTE;`
        i. After the above code, BYTE can be used to replace the words unsigned char
        ii. E.g **BYTE c1** is the same as **unsigned char c1**
15. What is the difference between typedef and #define?
    a. Typedef is used to give a data type a new name
        i. Obeys scope rule
    b. #define is use to define a special constant to a word
        i. Does not obey scope rule

Recursion

1. What is a recursive function?
    a. Is a divide and conquer method that solved the problem by reducing itself to smaller cases of the same problem
    b. Define as a routine that directly or indirectly call itself or call other functions
2. What is necessary for recursion?
    a. There must be a termination point
3. How are recursive functions implemented?
    a. Find the key step (recursive condition
    b. Find a stopping rule (terminating condition)
    c. Outline your algorithm (Use if-else)
    d. Check termination
4. How is recursion stopped when two or more functions are involved in indirect recursion?
    a. When the functions hits a terminating condition
5. Why must recursion stopped at a particular level?
    a. Every level have different arguments
6. Why is it necessary to stop recursion at some point?
    a. If there is no stop point, the recursion will loop infinitely
7. How does recursion simulate the operation of a loop?
    a. The recursion will keep on performing the same function until the terminating condition is met, just like how a loop will keep on repeating the same line of codes until the condition is false
8. What are similarities between iteration and recursion?
    a. Both execute a set of instructions until the the desired output is achieved
9. For defining a recursive function, what are possible causes for infinite recursion?
    a. When there are no terminating conditions
10. What for() statement performs the same job as fact() for computing factorial?
    a. `for(i=1;i<=num;i++)`

b.

```
f=f*i;
```