**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **18/06/2020** | **Name:** | **PRIYA P RAO** |
| **Course:** | **C Programming** | **USN:** | **4AL18EC041** |
| **Topic:** | * **Basic concepts in C** * **Conditionals and Loops** * **Functions, Arrays and Pointers** * **Strings and Function Pointers** | **Semester & Section:** | **4th sem ‘A’ section.** |
| **Github Repository:** | **Priya-Rao** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session**  **C:\Users\Pawan\Desktop\aaa.PNG**  **C:\Users\Pawan\Desktop\bbb.PNG**  **C:\Users\Pawan\Desktop\ccc.PNG** |
| **In today’s session I have learnt about:**  **Chapter 1: Basic concepts in C**   * **Introduction to C:**   **C is a general-purpose programming language that has been around for nearly 50 years. C has been used to write everything from operating systems to complex programs like the Python interpreter, Git, Oracle database, and more. The versatility of C is by design. It is a low-level language that relates closely to the way machines work while still being easy to learn.** Data Types:  C supports the following basic data types: int: integer, a whole number. float: floating point, a number with a fractional part. double: double-precision floating point value. char: single character.  * **Input and Output functions:**  Input: C supports a number of ways for taking user input.  getchar() Returns the value of the next single character input.   For example: **#include <stdio.h> int main()**  **{ char a = getchar(); printf("You entered: %c", a); return 0; }** Output: We have already used the printf() function to generate output in the previous lessons. In this lesson, we cover several other functions that can be used for output. putchar() Outputs a single character. For example: **#include <stdio.h> int main()**  **{ char a = getchar(); printf("You entered: "); putchar(a); return 0; }** Arithmetic Operators **C supports arithmetic operators + (addition), - (subtraction), \* (multiplication), / (division), and % (modulus division). Operators are often used to form a numeric expression such as 10 + 5, which in this case contains two operands and the addition operator. Numeric expressions are often used in assignment statements.**  **Chapter 2: Conditionals and Loops:**   * **Conditional Statements: if, else, switch. Conditional statements help you to make a decision based on certain conditions. These conditions are specified by a set of conditional statements having Boolean expressions which are evaluated to a Boolean value true or false.** * **Loops in C: In any programming language including C, loops are used to execute a set of statements repeatedly until a particular condition is satisfied.**   **Chapter 3: Functions, Array and Pointers:**   * **Functions: There are two types of functions in C programming:** * **Library Functions: are the functions which are declared in the C header files such as scanf (), printf(), gets(), puts(), ceil(), floor() etc.** * **User-defined functions: are the functions which are created by the C programmer, so that he/she can use it many times.** * **Array:**   **An array is a collection of data items, all of the same type, accessed using a common name. A one-dimensional array is like a list; A two dimensional array is like a table; The C language places no limits on the number of dimensions in an array, though specific implementations may.**   * **Pointers:**   **Pointers in C language is a variable that stores/points the address of another variable. A Pointer in C is used to allocate memory dynamically i.e. at run time. The pointer variable might be belonging to any of the data type such as int, float, char, double, short etc.**  **Chapter 4: Strings and Function Pointers:**   * **Strings:**   **In computer programming, a string is traditionally a sequence of characters, either as a literal constant or as some kind of variable. ... In formal languages, which are used in mathematical logic and theoretical computer science, a string is a finite sequence of symbols that are chosen from a set called an alphabet.**   * **Function Pointers:**   **Function Pointers point to code like normal pointers. In Functions Pointers, function's name can be used to get function's address. A function can also be passed as an arguments and can be returned from a function.** |