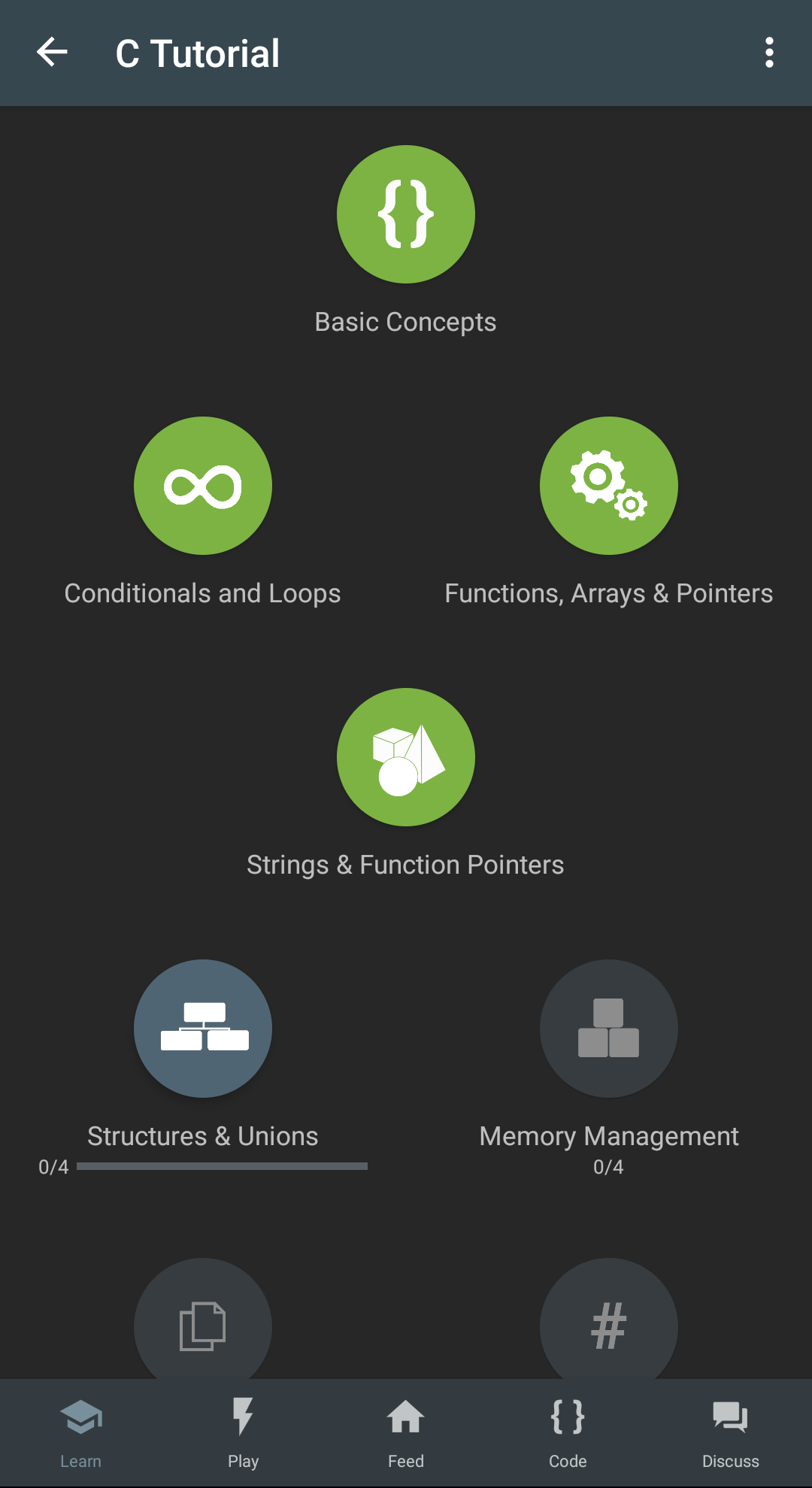
**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **18/06/2020** | **Name:** | **Lavanya B** |
| **Course:** | **C programming** | **USN:** | **4al17ec043** |
| **Topic:** | **Basic concepts**  **Conditional loops**  **Functions, array, pointers** | **Semester & Section:** | **6th A** |
| **Github Repository:** | **Lavanya-B** |  |  |

**Image of the session**

****

**Report**

**C programming**

**C is a general-purpose programming language. 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.**

**To print "Hello World"**

**#include <stdio.h>**

**int main() {**

**printf("Hello, World!\n");**

**return 0;**

**}**

**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.**

**Eg:**

**#include <stdio.h>**

**int main() {**

**printf("int: %ld \n", sizeof(int));**

**printf("float: %ld \n", sizeof(float));**

**printf("double: %ld \n", sizeof(double));**

**printf("char: %ld \n", sizeof(char));**

**return 0;**

**}**

**Input and Output**

**getchar() Returns the value of the next single character input.**

**gets() function is used to read input as an ordered sequence of characters, also called a string.**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int a;**

**scanf("%d", &a)**

**printf("You entered: %d", a)**

**return 0;**

**}**

**putchar() Outputs a single character.**

**puts() function is used to display output as a string. A string is stored in a char array.**

**Eg:**

**#include <stdio.h>**

**int main() {**

**char a[100];**

**gets(a);**

**printf("You entered: ");**

**puts(a);**

**return 0;**

**}**

**Comments**

**/\* multiple line comments\*/**

**//Single line comment**

**Operators**

**C supports arithmetic operators( \*,/, +, -)**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int x = 2;**

**x += 1;**

**x -= 1;**

**x \*= 3;**

**x /= 2;**

**x %= 2;**

**x += 3 \* 2;**

**printf("%d", x);**

**return 0;**

**}**

**Relational operators**

**< less than**

**<= less than or equal to**

**> greater than**

**>= greater than or equal to**

**== equal to**

**!= not equal to**

**Eg: #include <stdio.h>**

**int main() {**

**int in\_stock = 20;**

**if (in\_stock)**

**printf("Order received.\n"); }**

**Loops**

* **if statement**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int score = 89;**

**if (score > 75)**

**printf("You passed.\n");**

**return 0;**

**}**

* **Switch statement**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int num = 3;**

**switch (num) {**

**case 1:**

**printf("One\n");**

**break;**

**case 2:**

**printf("Two\n");**

**break;**

**case 3:**

**printf("Three\n");**

**break;**

**default:**

**printf("Not 1, 2, or 3.\n");**

**}**

**}**

* **While loop**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int count = 1;**

**while (count < 8) {**

**printf("Count = %d\n", count);**

**count++;**

**}**

**return 0;**

**}**

* **for loop**

**Eg:**

**#include <stdio.h>**

**int main() {**

**int i;**

**int max = 10;**

**for (i = 0; i < max; i++) {**

**printf("%d\n", i);**

**}**

**}**

**Functions**

**Functions are central to C programming and are used to accomplish a program solution as a series of subtasks.**

**By now you know that every C program contains a main() function. And you're familiar with the printf() function.**

**Eg:**

**#include <stdio.h>**

**/\* declaration \*/**

**int square (int num);**

**int main() {**

**int x, result;**

**x = 5;**

**result = square(x);**

**printf("%d squared is %d\n", x, result);**

**return 0;**

**}**

**/\* definition \*/**

**int square (int num) {**

**int y;**

**y = num \* num;**

**return(y);**

**}**

**Array**

**An array is a data structure that stores a collection of related values that are all the same type. The contents of an array are called elements with each element accessible by an index number. In C, index numbers start at 0.**

**Eg:**

**#include <stdio.h>**

**int main() {**

**float purchases[3] = {10.99, 14.25, 90.50};**

**float total = 0;**

**int k;**

**for (k = 0; k < 3; k++) {**

**total += purchases[k];**

**}**

**printf("Purchases total is %6.2f\n", total);**

**return 0;**

**}**

**Pointers**

**A pointer is a variable that contains the address of another variable. In other words, it "points" to the location assigned to a variable and can indirectly access the variable.**

**• Pointers should be initialized to NULL until they are assigned a valid location.**

**• Pointers can be assigned the address of a variable using the ampersand & sign.**

**• To see what a pointer is pointing to, use the \* again, as in \*p. In this case the \* is called the indirection or dereference operator. The process is called dereferencing.**

**#include <stdio.h>**

**int main() {**

**int x = 5;**

**int y;**

**int \*p = NULL;**

**p = &x;**

**y = \*p + 2;**

**y += \*p;**

**\*p = y;**

**(\*p)++;**

**printf("p is pointing to the value %d\n", \*p);**

**}**

**Strings**

**A string declaration can be made in several ways, each with its own considerations.**

**C provides the scanf(), gets(), and fgets() functions.**

**String output is handled with the fputs(), puts(), and printf() functions.**

**Eg:**

**#include <stdio.h>**

**int main()**

**{**

**char city[40];**

**printf("Enter your favorite city: ");**

**gets(city);**

**puts(city);**

**return 0;**

**}**