Day-4

I’ve installed the arch-linux (CLI Version only) successfully a few days ago. I’ll move forward by installing the DE but it’s not a big deal. What’s currently I’m focused is on editing the current DE elements and eventually create a full fledge custom DE. I’ve completed C language today as I’ll be using GTK (Gimp Toolkit) for UI designing and it’s primarily based upon C only. So, my current focus goes majority onto GTK with C and Figma for my own designs. And I'’ll keep learning about Linux on daily basis.

I already had the fundamental knowledge of programming from other languages already. So, it didn’t take me much time and notes to have a good hold on C.
In this document, I’ll discuss about the major topics I learnt in C which are enough for my GTK journey.

1) It has data type of mainly 3 types: int, float, char (understandable from their name itself).
2) While trying to print the variable of different type, we have their different notations used in *pritnf()* function. Like %d %f %c %s for example -
   ```
   printf(“This is for integer %d”,a);
   printf(“This is for float %f”,b);
   printf(“This is for char %c”, c);
   printf(“This is for string %s”,d);
   ```
3) The concept of operators is also same as other languages. But the increment and dicrement works a bit different here.
   ```
   printf(a++)        ----> It will first print the value of a and then increment it by one.
   printf(++a)        ----> It will first increment the value and then print the incremented value


   printf(a--)
   printf(--a)
   ```
   Same logic goes here for decrement also.
4) Now let’s look at how to take input from the users.
   We will use *scanf()* function for taking input and %d and other notations accordingly.
   Example -
   ```
   printf(“Enter the value \n”);
   scanf(“%d”, &inputData);
   ```
5) Let’s discuss type-casting. Conversion of one data type to another.
   Example -
   ```
   int num1 = 3, num2 = 4;
   pintf(“num1/num2 is %f”, (float) num1/num2);
   ```
6) Now we’re on the conditional statements and loops and functions just like any other programming language with different syntax. So, I think that would be too much time consuming for me to add structured code for basic knowledge. I’ll log only those things which I’ve learned something different from other programming languages, not just the syntax.
7) We now forward with the arrays. Declaring arrays in C are a bit different.
   For example -
   ```
   int arr[10] = {0,1,2,3,4,5,6,7,8,9};
   printf(“%d”, arr[7]);            -----> Output : 7
   ```

8) Now the pointers in C. Pointers are type of variable which store the memory address of other variables and elements. And we can change the address according to ourselves. We can use pointers to dynamically allocate the memory and make our program more memory efficient.
The syntax follows as -

```
int a = 98;
int* ptr;
ptr = &a;
*ptr = 188;
```

9) The strings also work a bit different in C. The string is an array of char data type and each letter as individual element of that array. And the last element in that array is always a null character. (‘\0’).
For example -

```
char name[3] = {‘m’, ‘y’, ‘\0’};
printf(“%s”, name);               ----> This gives output – my
```

10) Now the structures in C. Thery are nothing but works similar to OOP in other languages.
It will be more understandable by the syntax.

```
struct books{
      char name [50];
      char author [50];
      int price;
} book;

int main(){
     struct books bk1, bk2;
     strcpy(bk1.name, “Programming in C”);      ---> strcpy() is a function used to copy string
     strcpy(bk1.author, “Dennis”);
     bk1.price = 199;
     printStruct(bk1);
}


void printStruct(struct books bk){
         Printf(“Book’s name is %s\n”, bk.name);
         Printf(“Author’s name is %s\n”, bk.author);
        Printf(“The price of the book is %d”, bk.price);
}
```

Note: I’ve not mentioned all the topics and in very detail too. Cause I think my focus should be on GTK development rather than mastering one language. I learn the C language to the extent where I can easily get myself with GTK. And if there’s confusion in GTK or I've left some topics, I’ll study them on the go.