C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Variables, Data Types + Input/Output

(Chapter 1)

1. **First Program**

```c
#include<stdio.h>

int main() {
   printf("Hello World");
   return 0;
}
```

2. **Variables & Data Types + Constants & Keywords**

```c
#include<stdio.h>

int main() {
   int number;
   int age;
   int price;
   return 0;
}
```

```c
#include<stdio.h>

int main() {
   int age = 22;
   float pi = 3.14;
   char percentage = '%';
   return 0;
}
```

3. Comments

```c
#include<stdio.h>
//This program prints Hello World
int main() {
   printf("Hello World");
   return 0;
}
```

4. Output

```c
#include<stdio.h>

int main() {
   int age = 22;
   float pi = 3.14;
   char percentage = '%';


   printf("age is %d", age);
   printf("age is %f", pi);
   printf("age is %c", percentage);
   return 0;
}
```

5. Input (Sum of 2 numbers)

```c
#include<stdio.h>

int main() {
   int a, b;

   printf("enter a \n");
   scanf("%d", &a);

   printf("enter b \n");
```

```
    scanf("%d", &b);

    printf("sum of a & b is : %d \n", a+b);

    return 0;
}
```

6. **Practice Qs 1 (Area of Square)**

```
#include<stdio.h>
//area of square
int main() {
    int side;
    scanf("%d", &side);
    printf("%d", side * side);
    return 0;
}
```

7. **Practice Qs 2 (Area of Circle)**

```
#include<stdio.h>
//area of square
int main() {
    float radius;
    scanf("%f", &radius);
    printf("%f", 3.14 * radius * radius);
    return 0;
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Installation + Setup
Chapter 1 - Variables, Data types + Input/Output
Chapter 2 - Instructions & Operators
Chapter 3 - Conditional Statements
Chapter 4 - Loop Control Statements
Chapter 5 - Functions & Recursion
Chapter 6 - Pointers
Chapter 7 - Arrays
Chapter 8 - Strings
Chapter 9 - Structures
Chapter 10 - File I/O
Chapter 11 - Dynamic Memory Allocation

Instructions & Operators

(Chapter 2)

1. Type Declaration Instructions

```c
#include<stdio.h>

int main() {
   int age = 22;
   int oldAge = age;
   int newAge = oldAge + 2;
   printf("new age is : %d", newAge);

   int rupee = 1, dollar;
   dollar = 74;

   /*
       order of declaration is important - Wrong Declaration Order
       float pi = 3.14;
       float area = pi * rad * rad;
       float rad = 3;
   */
```

```
    // valid declaration
    int age1, age2, age3;
    age1 = age2 = age3 = 22;

    //invalid
    //int a1 = a2 = a3 = 22;

    return 0;
}
```

2. Arithmetic Instructions

```
#include<stdio.h>

int main() {
    int a = 1, b = 2, c = 3;
    //valid
    a = b + c;

    //invalid
    // b + c = a;

    printf("%d \n", 3 % 2);
    printf("%d \n", -3 % 2);
    return 0;
}
```

> Type Conversion

```
#include<stdio.h>

int main() {
    printf("sum of 2 & 3 : %d", 2 + 3);
    printf("sum of 2.0 & 3 : %f", 2.0 + 3);
    printf("sum of 2.0 & 3.0 : %f", 2.0 + 3.0);
    return 0;
}
```

> Associativity

```
#include<stdio.h>

int main() {
    printf(" Output : %d", 5+2/2*3);
```

```
    return 0;
}
```

3. Relational Operator

```
#include<stdio.h>

int main() {
    printf("%d \n", 4==4);

    printf("%d \n", 4<3);
    printf("%d \n", 3<4);
    printf("%d \n", 4<4);
    printf("%d \n", 4<=4);

    printf("%d \n", 4>3);
    printf("%d \n", 3>4);
    printf("%d \n", 4>4);
    printf("%d \n", 4>=4);

    printf("%d \n", 4!=4);
    printf("%d \n", 3!=4);
    return 0;
}
```

4. Logical Operator

```
#include<stdio.h>

int main() {
    printf("%d \n", 3<4 && 3<5);
    printf("%d \n", 3<4 && 5<4);

    printf("%d \n", 3<4 && 5<4);
    printf("%d \n", 3>4 && 5>4);
    printf("%d \n", 3<4 && 3<5);

    printf("%d \n", !(3<4 && 3<5));
    printf("%d \n", !(4<3 || 5<3));
    return 0;
}
```

5. Assignment Operator

```c
# include <stdio.h>

int main() {
   int a = 10;
   a += 10;
   printf("a+10 = %d \n", a);
   a -= 10;
   printf("a-10 = %d \n", a);
   a *= 10;
   printf("a*10 = %d \n", a);
   a /= 10;
   printf("a/10 = %d \n", a);
   a %= 10;
   printf("a%c10 = %d \n", '%', a);
   return 0;
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Conditional Statements

(Chapter 3)

1. **If-else**

```c
#include<stdio.h>

int main() {
   int age = 19;
   if(age >= 18) {
       printf("you are an adult");
   }
   else {
       printf("you are not an adult");
   }
   return 0;
}
```

> check if a number is odd or even

```c
#include<stdio.h>

int main() {
   int number;
```

```
    scanf("%d", &number);

    if(number % 2 == 0) {
        printf("even");
    }
    else {
        printf("odd");
    }
    return 0;
}
```

> Use of else if

```
#include<stdio.h>

int main() {
    int age;
    printf("Enter age : ");
    scanf("%d", &age);

    if(age < 12) {
        printf("child");
    }
    else if(age < 18) {
        printf("teenager");
    }
    else {
        printf("adult");
    }
    return 0;
}
```

2. Ternary Operator

```
#include<stdio.h>

int main() {
    int age;
    printf("Enter age : ");
    scanf("%d", &age);

    age > 18 ? printf("adult \n") : printf("not adult \n");

    int number = 7;
```

```
    int luckyNumber = 7;

    number == luckyNumber ? printf("you are lucky \n") : printf("you are not
lucky \n");

    return 0;
}
```

3. **Switch (integer)**

```
#include<stdio.h>
#include<math.h>

int main() {
    int day = 5;
    switch(day) {
        case 1 : printf("monday \n");
                 break;
        case 2 : printf("tuesday \n");
                 break;
        case 3 : printf("wednesday \n");
                 break;
        case 4 : printf("thursday \n");
                 break;
        case 5 : printf("friday \n");
                 break;
        case 6 : printf("saturday \n");
                 break;
        case 7 : printf("sunday \n");
                 break;
    }
    return 0;
}
```

4. **Switch (character)**

```
#include<stdio.h>
#include<math.h>

int main() {
    char day = 'f';
    switch(day) {
        case 'm' : printf("monday \n");
                 break;
```

```
        case 't' : printf("tuesday \n");
                   break;
        case 'w' : printf("wednesday \n");
                   break;
        case 'T' : printf("thursday \n");
                   break;
        case 'f' : printf("friday \n");
                   break;
        case 's' : printf("saturday \n");
                   break;
        case 'S' : printf("sunday \n");
                   break;
    }
    return 0;
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Installation + Setup
Chapter 1 - Variables, Data types + Input/Output
Chapter 2 - Instructions & Operators
Chapter 3 - Conditional Statements
Chapter 4 - Loop Control Statements
Chapter 5 - Functions & Recursion
Chapter 6 - Pointers
Chapter 7 - Arrays
Chapter 8 - Strings
Chapter 9 - Structures
Chapter 10 - File I/O
Chapter 11 - Dynamic Memory Allocation

Loop Control Statements

(Chapter 4)

1. **Syntax of 3 Loops**

```c
# include <stdio.h>

int main () {
   //for loop
   for(int i=1; i<=100; i++) {
       printf("%d\n", i);
   }

   //while loop
   int i=1;
   while(i<=100) {
       printf("%d\n", i);
       i++;
   }

   //do while loop
   i = 1;
   do {
```

```
        printf("%d\n", i);
        i++;
    } while(i<=100);

    return 0;
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Functions & Recursion

(Chapter 5)

1. **Function to print Hello**

```c
#include<stdio.h>

//function declaration/prototype
void printHello();

int main() {
   //function call
   printHello();
   return 0;
}

//function definition
void printHello() {
   printf("Hello!\n");
}
```

2. **Function to calculate square of a number**

```c
# include <stdio.h>
//function to calculate square of a number
int calcSquare(int n);

int main() {
   int n;
   printf("enter n : ");
   scanf("%d", &n);
   printf("square is : %d", calcSquare(n));
   return 0;
}

int calcSquare(int n) {
   return n * n;
}
```

3. **Function to calculate n factorial (using recursion)**

```c
# include <stdio.h>
//function to print factorial of n
int factorial(int n);

int main() {
   int n;
   printf("enter n : ");
   scanf("%d", &n);
   printf("factorial is : %d", factorial(n));
   return 0;
}

int factorial(int n) {
   if(n == 0) {
       return 1;
   }
   int factnm1 = factorial(n-1);
   int factn = factnm1 * n;
   return factn;
}
```

APNA COLLEGE

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Installation + Setup
Chapter 1 - Variables, Data types + Input/Output
Chapter 2 - Instructions & Operators
Chapter 3 - Conditional Statements
Chapter 4 - Loop Control Statements
Chapter 5 - Functions & Recursion
Chapter 6 - Pointers
Chapter 7 - Arrays
Chapter 8 - Strings
Chapter 9 - Structures
Chapter 10 - File I/O
Chapter 11 - Dynamic Memory Allocation

Pointers

(Chapter 6)

1. **Syntax**

```c
#include<stdio.h>

int main() {
   int age = 22;
   int *ptr = &age;
   int _age = *ptr;
   printf("%d\n", _age);

   //address
   printf("%p\n", &age);
   printf("%p\n", ptr);
   printf("%p\n", &ptr);

   //data
   printf("%d\n", age);
   printf("%d\n", *ptr);
   printf("%d\n", *(&age));
   return 0;
```

```
}
```

2. Pointers in Function call

```c
# include <stdio.h>

void square(int n);
void _square(int* n);

int main() {
   int number = 4;

   //call by value
   square(number);
   printf("n is : %d\n", number);

   //call by reference
   _square(&number);
   printf("n is : %d\n", number);
   return 0;
}

void square(int n) {
   n = n * n;
   printf("square is : %d\n", n);
}

void _square(int* n) {
   *n = *n * *n;
   printf("square is : %d\n", *n);
}
```

3. Swap 2 numbers

```c
# include <stdio.h>

void swap(int a, int b);
void _swap(int* a, int *b);

int main() {
   int x = 3, y = 5;

   //call by value
   swap(x, y);
```

```
    printf("x = %d & y = %d\n", x, y);

    //call by reference
    _swap(&x, &y);
    printf("x = %d & y = %d\n", x, y);
    return 0;
}

void swap(int a, int b) {
    int t = a;
    a = b;
    b = a;
}

void _swap(int* a, int* b) {
    int t = *a;
    *a = *b;
    *b = *a;
}
```

APNA COLLEGE

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Arrays

(Chapter 7)

1. **Syntax**

```c
# include <stdio.h>

int main() {
   int marks[3];
   printf("physics : ");
   scanf("%d", &marks[0]);

   printf("chem : ");
   scanf("%d", &marks[1]);

   printf("math : ");
   scanf("%d", &marks[2]);

   printf("physics = %d, ", marks[0]); //physics
   printf("chem = %d, ", marks[1]);  //chem
   printf("math = %d \n", marks[2]); //math
```

```
    return 0;
}
```

2. **Pointer Arithmetic**

```c
# include <stdio.h>

int main() {

    int age = 22;
    int *ptr = &age;

    int _age = 25;
    int *_ptr = &_age;

    printf("%u\n", ptr);
    ptr++;
    printf("%u\n", ptr);
    ptr--;
    printf("%u\n", ptr);
    ptr = ptr - _ptr;
    printf("%u\n", ptr);

    ptr = &_age;
    printf("%d\n", ptr == _ptr);

    return 0;
}
```

3. **Accessing an Array**

```c
# include <stdio.h>

void printNumbers(int *arr, int n);
void _printNumbers(int arr[], int n);

int main() {
    int arr[] = {1, 2, 3, 4, 5, 6};
    printNumbers(arr, 6);
    printNumbers(arr, 6);
    return 0;
}
```

```c
void printNumbers(int *arr, int n) {
    for(int i=0; i<n; i++) {
        printf("%d : %d\n", i, arr[i]);
    }
}

void _printNumbers(int arr[], int n) {
    for(int i=0; i<n; i++) {
        printf("%d : %d\n", i, arr[i]);
    }
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Installation + Setup
Chapter 1 - Variables, Data types + Input/Output
Chapter 2 - Instructions & Operators
Chapter 3 - Conditional Statements
Chapter 4 - Loop Control Statements
Chapter 5 - Functions & Recursion
Chapter 6 - Pointers
Chapter 7 - Arrays
Chapter 8 - Strings
Chapter 9 - Structures
Chapter 10 - File I/O
Chapter 11 - Dynamic Memory Allocation

Strings

(Chapter 8)

1. Strings

```c
# include <stdio.h>
# include <string.h>



int main() {
   //declaration
   char name[] = "Shradha Khapra";
   char course[] = {'a','p', 'n', 'a', ' ', 'c', 'o', 'l', 'l', 'e', 'g', 'e',
'\0'};

   //printing string
   for(int i=0; name[i] != '\0'; i++) {
      printf("%c", name[i]);
   }
   printf("\n");

   //printing string with pointer
   for(char *ptr=name; *ptr != '\0'; ptr++) {
      printf("%c", *ptr);
```

```
}
printf("\n");

//printing using format specifier
printf("%s\n", name);

//input a string
char firstName[40];
printf("enter first name : ");
scanf("%s", firstName);
printf("you first name is %s\n", firstName);
 char fullName[40];
printf("enter full name : ");
scanf("%s", fullName);
printf("you first name is %s\n", fullName);

// gets & puts
char fullName[40];
printf("enter full name : ");
fgets(fullName, 40, stdin);
puts(fullName);

//Library Functions
char name[] = "Shradha";
int length = strlen(name);
printf("the length of name : %d\n", length);

char oldVal[] = "oldValue";
char newVal[50];
strcpy(newVal, oldVal);
puts(newVal);

char firstStr[50] = "Hello ";
char secStr[] = "World";
strcat(firstStr, secStr);
puts(firstStr);

char str1[] = "Apple";
char str2[] = "Banana";
printf("%d\n", strcmp(str1, str2));

//enter String using %c
```

```
    printf("enter string : ");
    char str[100];
    char ch;
    int i = 0;

    while(ch != '\n') {
        scanf("%c", &ch);
        str[i] = ch;
        i++;
    }
    str[i] = '\0';
    puts(str);

    return 0;
}
```

> Some more Qs

```
# include <stdio.h>
# include <string.h>

// void printString(char arr[]);
// int countLength(char arr[]);
// void salting(char password[]);
// void slice(char str[], int n, int m);

//int countVowels(char str[]);

void checkChar(char str[], char ch);

int main() {
    char str[] = "ApnaCollege";
    char ch = 'x';
    checkChar(str, ch);
}

void checkChar(char str[], char ch) {
    for(int i=0; str[i] != '\0'; i++) {
        if(str[i] == ch) {
            printf("character is present!");
            return;
```

```
        }
    }
     printf("character is NOT present:(");
}




// int countVowels(char str[]) {
//     int count = 0;

//     for(int i=0; str[i] != '\0'; i++) {
//         if(str[i] == 'a' || str[i] == 'e' || str[i] == 'i' ||
//            str[i] == 'o' || str[i] == 'u') {
//                count++;
//            }
//     }
//     return count;
// }




// void slice(char str[], int n, int m) { // n & m are valid value
//     char newStr[100];
//     int j = 0;
//     for(int i=n; i<=m; i++, j++) {
//         newStr[j] = str[i];
//     }
//     newStr[j] = '\0';
//     puts(newStr);
// }



// void salting(char password[]) {
//     char salt[] = "123";
//     char newPass[200];

//     strcpy(newPass, password); // newPass = "test"
//     strcat(newPass, salt); // newPass = "test" + "123";
//     puts(newPass);
```

```
// }



// int countLength(char arr[]) {
//     int count = 0;
//     for(int i=0; arr[i]!='\0'; i++) {
//         count++;
//     }
//     return count-1;
// }

// void printString(char arr[]) {
//     for(int i=0; arr[i] != '\0' ;i++) {
//         printf("%c", arr[i]);
//     }
//     printf("\n");
// }
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Structures

(Chapter 9)

1. Structures

```c
# include <stdio.h>
# include <string.h>

struct student {
   char name[100];
   int roll;
   float cgpa;
};

typedef struct ComputerEngineeringStudent{
   int roll;
   float cgpa;
   char name[100];
} coe;

void printInfo(struct student s1);

int main() {
```

```
struct student s1;
// s1.name = "Shradha"; // not a modifiable value
strcpy(s1.name,"Shradha");
s1.roll = 64;
s1.cgpa = 9.2;

printf("student info : \n");
printf("name = %s\n", s1.name);
printf("roll no = %d\n", s1.roll);
printf("cgpa = %f\n", s1.cgpa);


//array of structures
struct student IT[60];
struct student COE[60];
struct student ECE[60];

//declaration
struct student s2 = {"Rajat", 1552, 8.6};
struct student s3 = {0};

printf("roll no of s2 = %d\n", s2.roll);
printf("roll no of s3 = %d\n", s3.roll);

//pointer to structure
struct student *ptr = &s1;
printf("student.name = %s\n", (*ptr).name);
printf("student.roll = %d\n", (*ptr).roll);
printf("student.cgpa = %f\n", (*ptr).cgpa);

//arrow operator
printf("student->name = %s\n", ptr->name);
printf("student->roll = %d\n", ptr->roll);
printf("student->cgpa = %f\n", ptr->cgpa);

//Passing structure to function
printInfo(s1);

//typedef keyword
coe student1;
student1.roll = 1664;
student1.cgpa = 6.7;
```

```
    strcpy(student1.name, "sudhir");

    return 0;
}

void printInfo(struct student s1) {
    printf("student info : \n");
    printf("name = %s\n", s1.name);
    printf("roll no = %d\n", s1.roll);
    printf("cgpa = %f\n", s1.cgpa);

    //change
    s1.roll = 1660; //but it won't be reflected to the main function
                    //as structures are passed by value
}
```

> Some more Qs

```
# include <stdio.h>
# include <string.h>

//user defined
typedef struct student {
    int roll;
    float cgpa;
    char name[100];
} stu ;

typedef struct computerengineeringstudent {
    int roll;
    float cgpa;
    char name[100];
} coe;

struct address {
    int houseNo;
    int block;
    char city[100];
    char state[100];
};

struct vector {
    int x;
```

```
    int y;
};

void calcSum(struct vector v1, struct vector v2, struct vector sum);

struct complex {
    int real;
    int img;
};

typedef struct BankAccount {
    int accountNo;
    char name[100];
} acc ;

int main() {
    acc acc1 = {123, "shradha"};
    acc acc2 = {124, "rajat"};
    acc acc3 = {125, "sudhir"};

    printf("acc no = %d", acc1.accountNo);
    printf("name = %s", acc1.name);
    return 0;
}

void calcSum(struct vector v1, struct vector v2, struct vector sum) {
    sum.x = v1.x + v2.x;
    sum.y = v1.y + v2.y;

    printf("sum of x is : %d\n", sum.x);
    printf("sum of y is : %d\n", sum.y);
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

File I/O

(Chapter 10)

```c
# include <stdio.h>

int main() {
   FILE *fptr;
   //Reading a file
   char ch;
   fptr = fopen("Test.txt", "r");
   if(fptr == NULL) {
       printf("doesn't exist!\n");
   } else {
       fscanf(fptr, "%c", &ch);
       printf("character in file is : %c\n", ch);
       fscanf(fptr, "%c", &ch);
       printf("character in file is : %c\n", ch);
       fclose(fptr);
   }

   //Writing in a file
   ch = 'M';
   fptr = fopen("NewFile.txt", "w");
```

```
    fprintf(fptr, "%cANGO", ch);
    fclose(fptr);

    //fgets
    fptr = fopen("NewFile.txt", "r");
    printf("character in file is : %c\n", fgetc(fptr));
    printf("character in file is : %c\n", fgetc(fptr));
    printf("character in file is : %c\n", fgetc(fptr));
    printf("character in file is : %c\n", fgetc(fptr));
    printf("character in file is : %c\n", fgetc(fptr));
    fclose(fptr);

    //fputc
    fptr = fopen("NewFile.txt", "w");
    fputc('a', fptr);
    fputc('p', fptr);
    fputc('p', fptr);
    fputc('l', fptr);
    fputc('e', fptr);
    fclose(fptr);

    //read the full file (EOF)
    fptr = fopen("NewFile.txt", "r");
    ch = fgetc(fptr);
    while(ch != EOF) {
        printf("%c", ch);
        ch = fgetc(fptr);
    }
    printf("\n");
    fclose(fptr);

    return 0;
}
```

C Language Tutorial

(Basic to Advanced)

Topics to be covered :

Dynamic Memory Allocation

(Chapter 11)

```c
# include <stdio.h>
# include <stdlib.h>
//Dynamic Memory Allocation

int main() {
   //sizeof function
   printf("%d\n", sizeof(int));
   printf("%d\n", sizeof(float));
   printf("%d\n", sizeof(char));

   //malloc
   // int *ptr;
   // ptr = (int *) malloc(5 * sizeof(int));

   // for(int i=0; i<5; i++) {
   //     scanf("%d", &ptr[i]);
   // }

   // for(int i=0; i<5; i++) {
   //     printf("number %d = %d\n", i+1, ptr[i]);
```

```
    // }

    //calloc
    int *ptr = (int *) calloc(5, sizeof(int));

    for(int i=0; i<5; i++) {
        printf("number %d = %d\n", i+1, ptr[i]);
    }

    //free
    free(ptr);

    return 0;
}
```