Functions:-

1) C Program to Demonstrate the Working of Keyword long

```
#include <stdio.h>
int main() {
   int a;
   long b;
   long long c;
   double e;
   long double f;
   printf("Size of int = %zu bytes \n", sizeof(a));
   printf("Size of long int = %zu bytes\n", sizeof(b));
   printf("Size of long long int = %zu bytes\n", sizeof(c));
   printf("Size of double = %zu bytes\n", sizeof(e));
   printf("Size of long double = %zu bytes\n", sizeof(f));
   return 0; }
```

2) C Program to Find the Sum of Natural Numbers using Recursion

```
#include <stdio.h>
int sum(int n);
int main() {
   int number, result;
   printf("Enter a positive integer: ");
   scanf("%d", &number);
   result = sum(number);
   printf("sum = %d", result);
   return 0; }
int sum(int n) {
   if (n != 0)
      return n + sum(n-1);
   else
      return n; }
```

3) C Program to Find G.C.D Using Recursion

```
#include <stdio.h>
int hcf(int n1, int n2);
int main() {
```

```
   int n1, n2;
   printf("Enter two positive integers: ");
   scanf("%d %d", &n1, &n2);
   printf("G.C.D of %d and %d is %d.", n1, n2, hcf(n1, n2));
   return 0;  }
int hcf(int n1, int n2) {
   if (n2 != 0)
      return hcf(n2, n1 % n2);
   else
      return n1;  }
```

4) C Program to Reverse a Sentence Using Recursion

```
#include <stdio.h>
void reverseSentence();
int main() {
   printf("Enter a sentence: ");
   reverseSentence();
   return 0;
}
void reverseSentence() {
   char c;
   scanf("%c", &c);
   if (c != '\n') {
      reverseSentence();
      printf("%c", c);
   }
}
```

5) C program to calculate the power using recursion

```
#include <stdio.h>
int power(int n1, int n2);
int main() {
   int base, a, result;
   printf("Enter base number: ");
   scanf("%d", &base);
   printf("Enter power number(positive integer): ");
```

```
    scanf("%d", &a);
    result = power(base, a);
    printf("%d^%d = %d", base, a, result);
    return 0;
}
int power(int base, int a) {
    if (a != 0)
        return (base * power(base, a - 1));
    else
        return 1;
}
```

Strings:-

1) C Program to Find the Frequency of Characters in a String

```
#include <stdio.h>
int main() {
    char str[1000], ch;
    int count = 0;
    printf("Enter a string: ");
    fgets(str, sizeof(str), stdin);
    printf("Enter a character to find its frequency: ");
    scanf("%c", &ch);
    for (int i = 0; str[i] != '\0'; ++i) {
        if (ch == str[i])
            ++count;
    }
    printf("Frequency of %c = %d", ch, count);
    return 0;
}
```

2) C Program to Remove all Characters in a String Except Alphabets

```
#include <stdio.h>
int main() {
  char line[150];
  printf("Enter a string: ");
  fgets(line, sizeof(line), stdin); // take input
```

```
  for (int i = 0, j; line[i] != '\0'; ++i) {
  while (!(line[i] >= 'a' && line[i] <= 'z') && !(line[i] >= 'A' && line[i] <= 'Z') &&
   !(line[i] == '\0')) {
       for (j = i; line[j] != '\0'; ++j) {
         line[j] = line[j + 1];
       }
       line[j] = '\0';
     }
  }
  printf("Output String: ");
  puts(line);
  return 0;
}
```

3) C Program to Find the Length of a String

```
#include <stdio.h>
int main() {
   char s[] = "Programming is fun";
   int i;
   for (i = 0; s[i] != '\0'; ++i);
   printf("Length of the string: %d", i);
   return 0;
}
```

4) C Program to Concatenate Two Strings

```
#include <stdio.h>
int main() {
 char s1[100] = "programming ", s2[] = "is awesome";
 int length, j;
 length = 0;
 while (s1[length] != '\0') {
  ++length;
 }
 for (j = 0; s2[j] != '\0'; ++j, ++length) {
  s1[length] = s2[j];
 }
```

```
  s1[length] = '\0';
  printf("After concatenation: ");
  puts(s1);
  return 0; }
```

5) C Program to Copy String Without Using strcpy()

```
#include <stdio.h>
int main() {
    char s1[100], s2[100], i;
    printf("Enter string s1: ");
    fgets(s1, sizeof(s1), stdin);
    for (i = 0; s1[i] != '\0'; ++i) {
        s2[i] = s1[i];  }
    s2[i] = '\0';
    printf("String s2: %s", s2);
    return 0;  }
```

Structures and Unios:-

1) C Program to Store Information of a Student Using Structure

```
#include <stdio.h>
struct student {
    char name[50];
    int roll;
    float marks;
} s;
int main() {
    printf("Enter information:\n");
    printf("Enter name: ");
    fgets(s.name, sizeof(s.name), stdin);
    printf("Enter roll number: ");
    scanf("%d", &s.roll);
    printf("Enter marks: ");
    scanf("%f", &s.marks);
    printf("Displaying Information:\n");
    printf("Name: ");
    printf("%s", s.name);
```

```
    printf("Roll number: %d\n", s.roll);
    printf("Marks: %.1f\n", s.marks);
    return 0;  }
```

2) C Program to Add Two Distances (in inch-feet system) using Structures

```
#include <stdio.h>
struct Distance {
  int feet;
  float inch;
} d1, d2, result;
int main() {
  printf("Enter 1st distance\n");
  printf("Enter feet: ");
  scanf("%d", &d1.feet);
  printf("Enter inch: ");
  scanf("%f", &d1.inch);
  printf("\nEnter 2nd distance\n");
  printf("Enter feet: ");
  scanf("%d", &d2.feet);
  printf("Enter inch: ");
  scanf("%f", &d2.inch);
  result.feet = d1.feet + d2.feet;
  result.inch = d1.inch + d2.inch;
  while (result.inch >= 12.0) {
    result.inch = result.inch - 12.0;
    ++result.feet;  }
  printf("\nSum of distances = %d\'-%.1f\"", result.feet, result.inch);
  return 0;  }
```

3) C Program to Add Two Complex Numbers by Passing Structure to a Function

```
#include <stdio.h>
typedef struct complex {
    float real;
    float imag;
} complex;
```

```
complex add(complex n1, complex n2);
int main() {
   complex n1, n2, result;
   printf("For 1st complex number \n");
   printf("Enter the real and imaginary parts: ");
   scanf("%f %f", &n1.real, &n1.imag);
   printf("\nFor 2nd complex number \n");
   printf("Enter the real and imaginary parts: ");
   scanf("%f %f", &n2.real, &n2.imag);
   result = add(n1, n2);
   printf("Sum = %.1f + %.1fi", result.real, result.imag);
   return 0;  }
complex add(complex n1, complex n2) {
   complex temp;
   temp.real = n1.real + n2.real;
   temp.imag = n1.imag + n2.imag;
   return (temp);  }
```

4) C Program to Store Information of Students Using Structure

```
#include <stdio.h>
struct student {
   char firstName[50];
   int roll;
   float marks;
} s[5];
int main() {
   int i;
   printf("Enter information of students:\n");
   for (i = 0; i < 5; ++i) {
      s[i].roll = i + 1;
      printf("\nFor roll number%d,\n", s[i].roll);
      printf("Enter first name: ");
      scanf("%s", s[i].firstName);
      printf("Enter marks: ");
```

```
        scanf("%f", &s[i].marks);  }
    printf("Displaying Information:\n\n");
    for (i = 0; i < 5; ++i) {
        printf("\nRoll number: %d\n", i + 1);
        printf("First name: ");
        puts(s[i].firstName);
        printf("Marks: %.1f", s[i].marks);
        printf("\n");  }
    return 0;  }
```

5) C Program to Store Data in Structures Dynamically

```
#include <stdio.h>
#include <stdlib.h>
struct course {
  int marks;
  char subject[30];
};
int main() {
  struct course *ptr;
  int noOfRecords;
  printf("Enter the number of records: ");
  scanf("%d", &noOfRecords);
  ptr = (struct course *)malloc(noOfRecords * sizeof(struct course));
  for (int i = 0; i < noOfRecords; ++i) {
    printf("Enter subject and marks:\n");
    scanf("%s %d", (ptr + i)->subject, &(ptr + i)->marks);
  }
  printf("Displaying Information:\n");
  for (int i = 0; i < noOfRecords; ++i) {
    printf("%s\t%d\n", (ptr + i)->subject, (ptr + i)->marks);
  }
  free(ptr);
  return 0;
}
```