1. Write a program to print N equal parts of a given string.

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

void equalsplit(char *str, int n)
{
   int len = strlen(str);
   if (len % n != 0)
   {
      printf("Unable to divide the string into equal parts.\n");
      return;
   }
   int size = len / n;
   int i, j;
   for (i = 0; i < len; i++)
   {
      if (i % size == 0 && i != 0)
         printf("\n"); // Start a new line for each part

      printf("%c", str[i]);
   }
   printf("\n");
}

int main()
{
   char str[100];
   int n;
   printf("Enter a string: ");
   scanf("%s", str);
   printf("Enter the number of equal parts: ");
   scanf("%d", &n);
   if (n <= 0)
   {
      printf("Invalid number of parts.\n");
      return 1;
   }
   printf("\nEqual parts of the string:\n");
   equalsplit(str, n);
   return 0;
}
```

Output:

```
Enter a string: DeeKayBee
Enter the number of equal parts: 3
Equal parts of the string:
Dee
Kay
Bee
```

2.Write a program to insert characters in a string at a certain position

```
#include <stdio.h>
#include <string.h>
void insert(char *str, char ch, int pos)
{
    int len = strlen(str);
    if (pos < 0 || pos > len)
    {
        printf("invalid attempt : out of bounds\n");
        return;
    }
    for (int i = len; i >= pos; i--)
    {
        str[i + 1] = str[i];
    }
    str[pos] = ch;
}
int main() {
    char str[100];
    char ch;
    int position;
    printf("Enter a string: ");
    scanf("%s", str);
    printf("Enter a character to insert: ");
    scanf(" %c", &ch);
    printf("Enter the position to insert: ");
    scanf("%d", &position);
    insert(str, ch, position);
    printf("\nAfter Insertion: %s\n", str);
    return 0;
}
```

Output:

```
Enter a string: deeaybee
Enter a character to insert: k
Enter the position to insert: 3
After Insertion: deekaybee
```

3. Write a Program to implement Anagram

```
#include <stdio.h>
#include <string.h>
void swap(char *x, char *y)
{
    char temp = *x;
    *x = *y;
    *y = temp;
}
void generate(char *str, int start, int end)
{
    if (start == end)
    {
        printf("%s\n", str);
    }
    else
    {
        for (int i = start; i <= end; i++) {
            swap(&str[start], &str[i]);
            generate(str, start + 1, end);
            swap(&str[start], &str[i]);
        }
    }
}
int main()
{
    char input[100];
    printf("Enter a string: ");
    scanf("%s", input);
    int length = strlen(input);
    printf("Anagrams of %s are:\n", input);
    generate(input, 0, length - 1);
    return 0;
}
```

Output:

```
Enter a string: DKB
Anagrams of DKB are:
DKB
DBK
KDB
KBD
BKD
BDK
```

4. Write a program in C to remove characters from a string except alphabets

```
#include <stdio.h>
#include <string.h>
void charremove(char *str)
{
    int len = strlen(str);
    int destIndex = 0;
    for (int srcIndex = 0; srcIndex < len; srcIndex++) {
        if ((str[srcIndex] >= 'a' && str[srcIndex] <= 'z') || (str[srcIndex] >= 'A' && str[srcIndex] <= 'Z'))
{
            str[destIndex] = str[srcIndex];
            destIndex++;
        }
    }
    str[destIndex] = '\0';
}
int main()
{
    char str[100];
    printf("Enter a string: ");
    scanf("%s", str);
    charremove(str);
    printf("\nString after removing non-alphabetic characters: %s\n", str);
    return 0;
}
```

Output:

```
Enter a string: D1K2B3
String after removing non-alphabetic characters: DKB
```

5. Write a program in C to find the frequency of characters

```
#include <stdio.h>
int main()
{
    char input[100];
    int freq[128] = {0};
    printf("Enter a string: ");
    gets(input);
    for (int i = 0; input[i] != '\0'; i++)
    {
        if (input[i] >= 0 && input[i] <= 127)
        {
            freq[input[i]]++;
        }
    }
    printf("Character frequencies:\n");
    for (int i = 0; i < 128; i++)
    {
        if (freq[i] > 0)
        {
            printf("'%c' : %d times\n", i, freq[i]);
        }
    }
    return 0;
}
```

Output:

```
Enter a string: Dhanwin
Character frequencies:
'D' : 1 times
'a' : 1 times
'h' : 1 times
'i' : 1 times
'n' : 2 times
'w' : 1 times
```

6. Write a program in C to check whether a character is a Hexadecimal Digit or not.

```
#include<stdio.h>
#include<ctype.h>
int main()
{
   char ch;
   printf("Enter a character: ");
   scanf( "%c", &ch );
   if(isdigit(ch))
      printf( "%c is a valid Hexadecimal Character",ch);
   else
      printf( "%c is not a valid Hexadecimal Character:",ch);
   return 0;
}
```

Output:

```
Enter a character: 8A
8 is a valid Hexadecimal Character
```

7. Write a program in C to replace the spaces in a string with a specific character.

```
#include <stdio.h>
#include <string.h>
int main()
{
   char str[25];
   char ch;
   printf("Enter the string:");
   gets(str);
   printf("Enter the character you wish to replace blank spaces with:");
   scanf("%c",&ch);
   for(int i = 0; i < strlen(str); i++)
   {
      if(str[i] == ' ')
         str[i] = ch;
   }
   printf("String after replacing spaces with given character: \n");
   printf("%s", str);
   return 0;
}
```

Output:

```
Enter the string:D K B
Enter the character you wish to replace blank spaces with:+
String after replacing spaces with given character:
D+K+B
```

8. Write a program in C to split strings by space into words

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str[100];
    char newstr[10][10];
    int i,j,k;
    printf("Enter a string:");
    gets(str);
    j=0; k=0;
    for(i=0;i<=(strlen(str));i++)
    {
        if(str[i]==' '||str[i]=='\0')
        {
            newstr[k][j]='\0';
            k++;
            j=0;
        }
        else
        {
            newstr[k][j]=str[i];
            j++;
        }
    }
    printf("\n After split by space:\n");
    for(i=0;i < k;i++)
        printf(" %s\n",newstr[i]);
    return 0;
}
```

Output:

```
Enter a string:I am Dhanwin
After split by space:
 I
 am
 Dhanwin
```

9. Write a C program to reverse all the vowels present in a string. Return the newly created string.

```
#include <stdio.h>
#include <string.h>
int vowel(char ch)
{
    switch (ch)
    {
        case 'a': case 'e': case 'i': case 'o': case 'u':
        case 'A': case 'E': case 'I': case 'O': case 'U':
            return 1;
        default:
            return 0;
    }
}
void reverse(char *str)
{
    int l = 0;
    int r = strlen(str) - 1;
    while (l < r)
    {
        while (l < r && !vowel(str[l]))
        {
            l++;
        }
        while (l < r && !vowel(str[r]))
        {
            r--;
        }
        if (l < r)
        {
            char temp = str[l];
            str[l] = str[r];
            str[r] = temp;
            l++;
            r--;
        }
    }
}
int main() {
    char input[100];
    printf("Enter a string: ");
    scanf("%s", input);
    reverse(input);
    printf("String with reversed vowels: %s\n", input);
    return 0;
}
```

Output:

```
Enter a string: Dhinwan
String with reversed vowels: Dhanwin
```

10. Write a C program to find the longest palindromic substring from a given string. Return the substring.

```
#include <stdio.h>
#include <string.h>
int palindrome(char *s, int begin, int end)
{
    while (begin < end)
    {
        if (s[begin] != s[end])
        {
            return 0;
        }
        begin++;
        end--;
    }
    return 1;
}
char* longest(char *s)
{
    int len = strlen(s);
    if (len <= 1)
    {
        return s;
    }
    int max = 1;
    int begin = 0;
    for (int i = 0; i < len; i++)
    {
        for (int j = i + 1; j < len; j++)
        {
            if (j - i + 1 > max && palindrome(s, i, j))
            {
                max = j - i + 1;
                begin = i;
            }
        }
    }
    char *res = malloc((max + 1) * sizeof(char));
    strncpy(res, s + begin, max);
    res[max] = '\0';
    return res;
}
int main()
{
    char input[100];
    printf("Enter a string: ");
    scanf("%s", input);
    char *result = longest(input);
    printf("Longest Palindromic Substring: %s\n", result);
    free(result);
```

```
    return 0;
}
```

Output:

```
Enter a string: DhanwinracecarKB
Longest Palindromic Substring: racecar
```