1. Compare dates and write Early/ Same / Late

```
#include <stdio.h>
#include<string.h>
#include<stdlib.h>
char compare(char time1[],char time2[]){
        int d1,m1,y1;
        int d2,m2,y2;
        sscanf(time1,"%d/%d/%d",&d1,&m1,&y1);
        sscanf(time2,"%d/%d/%d",&d2,&m2,&y2);
        if(d1<d2||(d1==d2 && (m1<m2||(m1==m2 && y1<y2)))){
                return 'E';
        }
        else if(d1==d2&&m1==m2&&y1==y2){
                return 'S';
        }
        else{
                return 'L';
}}
int main(){
        int n;
        printf("enter the length: ");
        scanf("%d",&n);
        char str[100];
        for(int i=0;i<n;i++){
                char t1[1000];
                char t2[1000];
                scanf("%s %s",t1,t2);
                str[i]=compare(t1,t2);
        }
        for(int i=0;i<n;i++){
                printf("%c\n",str[i]);
        }
        return 0;
}
```

INPUT:

```
2
12/08/24 10/09/23
12/09/22 12/09/22
```

OUTPUT:

```
L
S
```

2. String add (velocity)

```
#include <stdio.h>
#include<string.h>
 int main()
{
    char str[1000];
    char sub_string[100];
    int position;
    fgets(str,1000,stdin);
    str[strlen(str)-1]='\0';
    fgets(sub_string,100,stdin);
    sub_string[strlen(sub_string)-1]='\0';
    scanf("%d",&position);
    char result[1000];
    strncpy(result,str,position);
    result[position]='\0';
    strcat(result,sub_string);
    strcat(result,str+position);
    printf("%s",result);
 return 0;
}
```

INPUT:

```
Abhsh
ila
3
```

OUTPUT:

```
Abhilash
```

3. Non-anagrams

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
bool isAnagrams(char str1[],char str2[]){
        if(strlen(str1)!=strlen(str2)){
          return false;
        }
        int hash[256]={0};
        for(int i=0;i<strlen(str1);i++){
          hash[str1[i]-'a']++;
          hash[str2[i]-'a']--;
        }
        for(int i=0;i<256;i++){
          if(hash[i]!=0){
                    return false;
          }
        }
        return true;
}
void isUnique(char ans[][100], int m){
        for(int i=0;i<m;i++){
          int flag=0;
          for(int j=0;j<m;j++){
                    if(i==j){
                              continue;
                    }
                    if(isAnagrams(ans[i],ans[j])){
                              flag=1;
                              break;
                    }
          }
          if(flag==0){
                    printf("%s ",ans[i]);
          }
        }
}
int main(){
   char str[1000];
   fgets(str,1000,stdin);
   str[strlen(str)-1]='\0';
   int n=strlen(str);
   char ans[100][100];
   int count=0;
   char *token = strtok(str," ");
   while(token != NULL){
          strcpy(ans[count++],token);
          token = strtok(NULL," ");
   }
   isUnique(ans,count);
   return 0;
}
```

INPUT:
one two three four two neo

OUTPUT:
three four

4. Input str1, st2 check in str3

```
#include <stdio.h>
#include <string.h>
int main(){
        char str1[100],str2[100],str3[100];
        int hash[256]={0};
        int n,m,l;
        int flag=1;
        fgets(str1,100,stdin);
        fgets(str2,100,stdin);
        fgets(str3,100,stdin);
        str1[strlen(str1)-1]='\0';
        str2[strlen(str2)-1]='\0';
        str3[strlen(str3)-1]='\0';
        n=strlen(str1);
        m=strlen(str2);
        l=strlen(str3);
        for(int i=0;i<n;i++){
          hash[str1[i]-'a']++;
        }
        for(int i=0;i<m;i++){
          hash[str2[i]-'a']++;
        }
        for(int i=0;i<l;i++){
          hash[str3[i]-'a']--;
        }
        for(int i=0;i<26;i++){
          if(hash[i]!=0){
                    flag=0;
          }
        }
        if(flag==1){
          printf("string3 has both string1 and string2\n");
        }
        else{
          printf("string1 and string2 are not equal to string3\n");
        }

   return 0;
}
```

<u>INPUT</u>:

```
abcd
efgh
```

<u>OUTPUT</u>:

```
abcdefgh
```

5. INPUT: aaabhic   OUTPUT: bhic

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

int main(){
        char a[20];
        printf("enter the string: ");
        scanf("%s",a);

        char prev=a[0];
        int n=strlen(a);
        for(int i=1;i<n;i++){
                if(a[i] == prev){
                        while(i<n && a[i]==prev)
                {
                i++;
        }
        prev = a[i];
        }
        else{
                printf("%c",prev);
                prev=a[i];
        }
}
if(a[n-1]!=a[n-2]){
    printf("%c",prev);
}
return 0;
}
```

6. Vowels

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

int main()
{
    char str[100];
    int m;
    int totalvowels=0,vowels=0,ans=0;
    printf("enter string\n");
    fgets(str,99,stdin);
    str[strlen(str)-1] = '\0';
    m=strlen(str);
    for(int i=0;i<m;i++)
    {
        if( str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u')
        {
            totalvowels++;
        }
    }
    for(int i=0;i<m-1;i++)
    {
        if( str[i] == 'a' || str[i] == 'e' || str[i] == 'i' || str[i] == 'o' || str[i] == 'u')
        {
            vowels++;
        }
        if(vowels>totalvowels-vowels)
        {
            ans++;
        }
    }
    printf("%d",ans);

    return 0;
}
```

INPUT:

```
cprogram
```

OUTPUT:

```
1
```

7. Maximum Operations (Skeleton will be given)

```
int maximumOperations(char* s) {
    int n = strlen(s);
    int max_operations = 0;

    for (int i = 0; i < n - 1; i++) {
        // Check if characters are lexicographically consecutive
        if (s[i] + 1 == s[i + 1] || s[i] - 1 == s[i + 1]) {
            max_operations++;
            // Remove the consecutive characters from the string
            for (int j = i; j < n - 2; j++) {
                s[j] = s[j + 2];
            }
            n -= 2; // Update the length of the string after removal
            i = -1; // Reset i to check from the beginning
        }
    }

    return max_operations;
}
```

8. Decending order score(STRUCTURE)

```
#include <stdio.h>
typedef struct{
        int roll;
        int score;
}
record;
 void bubbleSort(record records[],int n){
        int i,j;
        for(i=0;i<n-1;i++){
                for(j=0;j<n-i-1;j++){
                        if(records[j].score<records[j+1].score){
                                record temp=records[j];
                                records[j]=records[j+1];
                                records[j+1]=temp;
                        }
                }
        }
}
int main(){
        int n=5,i,j;
        record records[n];
        printf("enter the records (roll-score):\n");
        for(i=0;i<n;i++){
                scanf("%d-%d",&records[i].roll,&records[i].score);
        }
        bubbleSort(records,n);
        for(i=0;i<n;i++){
                int flag=0;
                for(j=0;j<i;j++){
                        if(records[j].roll == records[i].roll){
                                flag=1;
                                break;
                        }
                }
                if(flag==0){
                        printf("%d-%d\n",records[i].roll,records[i].score);
                }
        }
return 0;
}
```

<u>INPUT</u>:

```
        1001-89
        1002-35
        1003-56
        1003-45
        1002-29
```

<u>OUTPUT</u>:

```
        1001-89
        1003-56
        1002-35
```

9. Monitor Array

```
#include <stdio.h>
int main()
{
   int n,i;
   int sum=0;
   scanf("%d",&n);
   int a[n];
   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
   for(i=0;i<n;i++)
   {
      sum=sum+a[i];
   }
   int total=(n*(n+1))/2;
   int ans=total-sum;
   printf("%d",ans);
   return 0;
}
```

INPUT:

```
            5
            0 2 3 4 5
```

OUTPUT:

```
            1
```

10. Sort comma separated strings Alphabetically

INPUT:
aaab, bcda, abcd
OUTPUT:
abcd, bcda, aaab

```
#include <stdio.h>
#include <string.h>
#define MAX_SIZE 100
#define MAX_WORD_SIZE 20

int compareStrings(const char* a, const char* b) {
    return strcmp(a, b);
}
void bubbleSort(char words[][MAX_WORD_SIZE], int count) {
    for (int i = 0; i < count - 1; i++) {
        for (int j = 0; j < count - i - 1; j++) {
            if (compareStrings(words[j], words[j + 1]) > 0) {
                // Swap words[j] and words[j + 1]
                char temp[MAX_WORD_SIZE];
                strcpy(temp, words[j]);
                strcpy(words[j], words[j + 1]);
                strcpy(words[j + 1], temp);
            }
        }
    }
}
int main() {
    char input[MAX_SIZE];
    char words[MAX_SIZE][MAX_WORD_SIZE];
    int count = 0;

    // Get input from the user
    printf("Enter a comma-separated sequence of words: ");
    fgets(input, MAX_SIZE, stdin);

    // Remove newline character from input
    input[strcspn(input, "\n")] = 0;

    // Split the input into a list of words
    char* token = strtok(input, ",");
    while (token != NULL && count < MAX_SIZE) {
        strncpy(words[count++], token, MAX_WORD_SIZE);
        token = strtok(NULL, ",");
    }

    // Sort the array of words alphabetically using bubble sort
    bubbleSort(words, count);

    // Print the sorted words as a comma-separated sequence
    for (int i = 0; i < count; i++) {
        printf("%s", words[i]);
        if (i < count - 1) {
            printf(",");
        }
    }

    return 0;
}
```

11. Find position of char and string in main string

```
#include <stdio.h>
#include<string.h>
int find(char *str,char c,char *sub)
{
   int a=strchr(str,c)?strchr(str,c)-str:-1;
   int b=strstr(str,sub)?strstr(str,sub)-str:-1;

   return a+b;
}
int main()

{
   char str[100];
   fgets(str,100,stdin);
   char c;
   scanf("%c",&c);
   char sub[20];
   scanf("%s",sub);
   int a=find(str,c,sub);
   printf("%d",a);
   return 0;
}
```

INPUT:

```
this is a program
h
program
```

OUTPUT:

```
11
```

12. Merge Sort List

```
#include <stdio.h>
int main()
{
    int n1, n2;
    scanf("%d", &n1);
    scanf("%d", &n2);
    int list1[n1], list2[n2];
    for(int i=0;i<n1;i++){
        scanf("%d", &list1[i]);
    }
    for(int i=0;i<n2;i++){
        scanf("%d", &list2[i]);
    }
    int mergedSize=n1+n2;
    int mergedArr[mergedSize];
    for(int i=0;i<n1;i++){
        mergedArr[i]=list1[i];
    }
    for(int i=0;i<n2;i++){
        mergedArr[n1+i]=list2[i];
    }
    for(int i=0;i<mergedSize-1;i++){
        for(int j=i+1;j<mergedSize;j++){
            if(mergedArr[i]<mergedArr[j]){
                int temp=mergedArr[i];
                mergedArr[i]=mergedArr[j];
                mergedArr[j]=temp;
            }
        }
    }
    for(int i=0; i<mergedSize;i++){
        printf("%d\t", mergedArr[i]);
    }

    return 0;
}
```

INPUT:

```
3
4
1 3 9
5 2 7 8
```

OUTPUT:

```
9 8 7 5 3 2 1
```