PROGRAMS FOR PLACEMENT – SET 3 SOLUTIONS

1)

```c
/*
program to copy a string
*/

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

int main()
{
    char str1[50];char str2[50];
    printf("enter string: ");
    fgets(str1,sizeof(str1),stdin);
    strcpy(str2,str1);
    printf("copied String: %s",str2);
    return 0;
}
```

2)

```c
/*
    program to reverse string
*/
#include<stdio.h>
#include<string.h>

int main()
{
    char str1[50]={0},str2[50]={0}; /////initialize string to null to avoid garbage
    int i;
    printf("enter string: ");
    fgets(str1,sizeof(str1),stdin);
    int n = strlen(str1)-1; ////// neglect '/n' in string str1
    for(i=0;i<=n;i++)
    {
        str2[i] = str1[n-i]; ///////logic to reverse string
    }
    printf("Reverse String is: %s\n",str2);
    return 0;
}
```

3)

```c
/*
    program to concatenate a string
*/
#include<stdio.h>
#include<string.h>

int main()
{
    char str1[25] = {0};
    char str2[50] = {0};


    printf("Enter String1: ");
    scanf("%s",str1);

    printf("Enter String2: ");
    scanf("%s",str2);

    strcat(str1,str2); ////concatenate string 2 to string 1

    printf("concatenated string is: %s\n",str1);

    return 0;
}
```

4)

```c
/*
    program to print a string
*/
#include<stdio.h>
#include<string.h>

int main()
{
    char str1[25] = {0};


    printf("Enter String1: ");
    fgets(str1,sizeof(str1),stdin);


    printf("string is: %s\n",str1);

    return 0;
}
```

5)

```c
/*
    program to print a string
*/
#include<stdio.h>
#include<string.h>

int main()
{
    char str1[25] = {0};


    printf("Enter String1: ");
    fgets(str1,sizeof(str1),stdin);


    printf("string length is: %ld\n",strlen(str1)-1); ///-1 to remove the end of line character(\n)

    return 0;
}
```

6)

```c
/*
    program to concatenate a string
*/
#include<stdio.h>
#include<string.h>

int main()
{
    char str1[25] = {0};
    char str2[50] = {0};


    printf("Enter String1: ");
    scanf("%s",str1);

    printf("Enter String2: ");
    scanf("%s",str2);

    int val = strcmp(str1,str2); ////compare string 2 to string 1

    if(val==0)
    {
        printf("two strings are same\n");
    }
    else
    {
        printf("two string are not same\n");
    }

    return 0;
```

7)

```c
/*
Write a Program to print Length of the string without using strlen() function
*/
#include<stdio.h>
#include<string.h>
int main()
{
    char str1[50]={0};
    char c;
    int i=0;
    printf("enter string: ");
    fgets(str1,sizeof(str1),stdin);

    while(1)
    {
        c = str1[i];
        if(c=='\n')
        {
            break;
        }
        i++;
    }
    printf("length of string is: %d\n",i);
    return 0;
}
```

8)

```c
#include <stdio.h>

int main()
{
    char    str[100];
    int     counter;

    printf("Enter a string: ");
    fgets(str,sizeof(str),stdin);

    // toggle each string characters
    for(counter=0;str[counter]!='\n';counter++)
    {
        if(str[counter]>='A' && str[counter]<='Z')
            str[counter]=str[counter]+32;   //convert into lower case
        else if(str[counter]>='a' && str[counter]<='z')
            str[counter]=str[counter]-32;   //convert into upper case
    }

    printf("String after toggle each characters: %s",str);
    return 0;
}
```

9)

```c
/*Program to remove Vowels from String */

#include<stdio.h>
#include<string.h>
void main()
{
    char str[20];
    int len, i, j;
    printf("Enter a string : ");
    fgets(str,sizeof(str),stdin);
    len=strlen(str);
    for(i=0; i<len; i++)
    {
        if(str[i]=='a' || str[i]=='e' || str[i]=='i' ||
        str[i]=='o' || str[i]=='u' || str[i]=='A' ||
        str[i]=='E' || str[i]=='I' || str[i]=='O' ||
        str[i]=='U')
        {
            for(j=i; j<len; j++)
            {
                str[j]=str[j+1];
            }
        len--;
        }
    }
    printf("New string is : %s",str);
}
```

10)

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

// A function to check if a string str is palindrome
void isPalindrome(char str[])
{
    // Start from leftmost and rightmost corners of str
    int l = 0;
    int h = strlen(str) - 1;

    // Keep comparing characters while they are same
    while (h > l)
    {
        if (str[l++] != str[h--])
        {
            printf("%s is Not Palindrome", str);
            return;
        }
    }
    printf("%s is palindrome", str);
}

// Driver program to test above function
int main()
{
    isPalindrome("abba");
    isPalindrome("wow");
    isPalindrome("hello");
    return 0;
}
```

11)

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

int main()
{
  char ch, input[100], output[100];
  int no[26] = {0}, n, c, t, x;

  printf("Enter some text\n");
  scanf("%s", input);

  n = strlen(input);

  /** Storing how many times characters (a to z)
    appears in input string in an array */

  for (c = 0; c < n; c++)
  {
    ch = input[c] - 'a';
    no[ch]++;
  }

  t = 0;

  /** Insert characters 'a' to 'z' in output string as many times
    as they appear in the input string */

  for (ch = 'a'; ch <= 'z'; ch++)
  {
    x = ch - 'a';

    for (c = 0; c < no[x]; c++)
    {
      output[t] = ch;
      t++;
    }
  }
  output[t] = '\0';

  printf("%s\n", output);

  return 0;
}
```

12)

```
// Program to remove brackets from an algebraic string

#include<stdio.h>
int main()
{
      int i=0,c=0,j=0;
      char a[100],b[100];

      printf("\nEnter the string : ");
      scanf("%s",a);
      while(a[i]!='\0')
      {
            if((a[i]=='(') && (a[i-1]=='-'))
            {
                  if(c==0)
                  {j=i;}
                  else
                  {j=c;}
                  while(a[i]!=')')
                  {
                        if(a[i+1]=='+')
                        b[j++]='-';
                        else if(a[i+1]=='-')
                        b[j++]='+';
                        else if(a[i+1]!=')')
                        b[j++]=a[i+1];
                        i++;
                  }
                  c=j+1;
            }
            else if(a[i]=='(' && a[i-1]=='+')
            {
                  if(c==0)
                  {j=i;}
                  else
                  {j=c;}
                  while(a[i]!=')')
                  {
                        b[j++]=a[i+1];
                        i++;
                  }
                  j--;
                  c=j+1;
            }
            else if(a[i]==')')
            {
            i++;
            continue;
            }
```

```
            else
            {
                  b[j++]=a[i];
            }
                  i++;
            }
            b[j]='\0';
      printf("%s",b);
      return 0;
}
```

13)

```c
/*
    Program to Remove Characters in String Except Alphabets
*/
#include<stdio.h>

int main()
{
    char line[150];
    int i, j;
    printf("Enter a string: ");
    fgets(line,sizeof(line),stdin);

    for(i = 0; 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;
}
```

14)

```c
/*
    Program to remove Whitespaces from String
*/
#include <stdio.h>

int main()
{
   char text[100], blank[100];
   int c = 0, d = 0;

   printf("Enter some text\n");
   fgets(text,sizeof(text),stdin);

   while (text[c] != '\0')
   {
      if (!(text[c] == ' ')) {
        blank[d] = text[c];
        d++;
      }
      c++;
   }

   blank[d] = '\0';

   printf("Text after removing blanks\n%s\n", blank);

   return 0;
}
```

15)

```c
#include <stdio.h>
void main()
{
    char string[80];
    int count, nc = 0, sum = 0;

    printf("Enter the string containing both digits and alphabet \n");
    scanf("%s", string);
    for (count = 0; string[count] != '\0'; count++)
    {
        if ((string[count] >= '0') && (string[count] <= '9'))
        {
            nc += 1;
            sum += (string[count] - '0');
        }
    }
    printf("NO. of Digits in the string = %d\n", nc);
    printf("Sum of all digits = %d\n", sum);
}
```

16)

```c
#include <stdio.h>
#define MAX 100

int main()
{
    char str[MAX]={0};
    int i;

    printf("Enter a string: ");
    scanf("%[^\n]s",str); //read string with spaces

    for(i=0; str[i]!='\0'; i++)
    {
        //check first character is lowercase alphabet
        if(i==0)
        {
            if((str[i]>='a' && str[i]<='z'))
                str[i]=str[i]-32; //subtract 32 to make it capital
            continue; //continue to the loop
        }
        if(str[i]==' ')//check space
        {
            //if space is found, check next character
            ++i;
            //check next character is lowercase alphabet
            if(str[i]>='a' && str[i]<='z')
            {
                str[i]=str[i]-32; //subtract 32 to make it capital
                continue; //continue to the loop
            }
        }
        else
        {
            //all other uppercase characters should be in lowercase
            if(str[i]>='A' && str[i]<='Z')
                str[i]=str[i]+32; //subtract 32 to make it small/lowercase
        }
    }

    printf("Capitalize string is: %s\n",str);

    return 0;
}
```

17)

```c
#include<stdio.h>

int main()
{
   char string[100];
   int c = 0, count[26] = {0}, x;

   printf("Enter a string\n");
   fgets(string,sizeof(string),stdin);

   while (string[c] != '\0') {
   /** Considering characters from 'a' to 'z' only and ignoring others. */

      if (string[c] >= 'a' && string[c] <= 'z') {
         x = string[c] - 'a';
         count[x]++;
      }

      c++;
   }

   for (c = 0; c < 26; c++)
         printf("%c occurs %d times in the string.\n", c + 'a', count[c]);

   return 0;
}
```

18)

```c
#include<stdio.h>

int main()
{
   char string[100];
   int c = 0, count[26] = {0}, x;

   printf("Enter a string\n");
   fgets(string,sizeof(string),stdin);

   while (string[c] != '\0') {
   /** Considering characters from 'a' to 'z' only and ignoring others. */

      if (string[c] >= 'a' && string[c] <= 'z') {
         x = string[c] - 'a';
         count[x]++;
      }

      c++;
   }

   for (c = 0; c < 26; c++)
         {
            if(count[c]==1)
            {
                printf("%c, ",c+'a');
            }
         }

   return 0;
}
```

19)

```c
#include <stdio.h>

int check_anagram(char [], char []);

int main()
{
  char a[100], b[100];

  printf("Enter a string\n");
  fgets(a,sizeof(a),stdin);

  printf("Enter a string\n");
  fgets(b,sizeof(b),stdin);

  if (check_anagram(a, b) == 1)
    printf("The strings are anagrams.\n");
  else
    printf("The strings aren't anagrams.\n");

  return 0;
}

int check_anagram(char a[], char b[])
{
  int first[26] = {0}, second[26] = {0}, c=0;

  // Calculating frequency of characters of first string

  while (a[c] != '\0')
  {
    first[a[c]-'a']++;
    c++;
  }

  c = 0;

  while (b[c] != '\0')
  {
    second[b[c]-'a']++;
    c++;
  }

  // Comparing frequency of characters

  for (c = 0; c < 26; c++)
  {
    if (first[c] != second[c])
      return 0;
  }

  return 1;
}
~
```

20)

```
void main()
{
        char a[100],b[100],c[100],d[100];
        int i,j,k,l1,l2,l3,m,count=-1;
        printf("\t\t\t   REPLACE OF STRING ");
        printf("\n\t\t\t**********************");
        printf("\n\n\t INPUT");
        printf("\n\t*******");
        printf("\n\nEnter the First String : ");
        scanf("%[a-z ]s",a);
        printf("\nEnter the Substring : ");
        scanf("%s",b);
        printf("\nEnter the Replace String : ");
        scanf("%s",c);
        printf("\n\n\tOUTPUT");
        printf("\n\t*******");
        for(i=0;a[i]!='\0';i++)
             l1=i;
             for(i=0;b[i]!='\0';i++)
                  l2=i;
                  for(i=0;c[i]!='\0';i++)
                       l3=i;
                       for(i=0;a[i]!='\0';i++)
                            d[i]=a[i];
                            d[i]='\0';
                            if(l1>=l2)
                             {
                                do
                                  {
                                      for(i=0,j=0;a[i]!='\0' && b[j]!='\0';i++ )
                                          {
                                                if(a[i]==b[j])
                                                  {
                                                     count++;
                                                     j++;
                                                  }
                                                 else
                                                  {
                                                         count=-1;
                                                          if(j>0)
                                                             i--;
                                                             j=0;
                                                  }
                                          }
                                                     m=i;
                                              if(count==l2)
                                                {
                                                     i=i-j;
                                                     for(k=0;c[k]!='\0';i++,k++)
                                                           a[i]=c[k];
                                                           if(l2!=l3)
                                                            {
                                                                for( ;d[m]!='\0';m++,i++)
                                                                     {
                                                                         a[i]=d[m];
                                                                     }
                                                            }
                                                                     //a[i]='\0';
                                                                     //printf("\nReplaced string:%s",a);
                                                }
                                                                      else
                                                                      break;
                                                                      //printf("\nThe substring is not there");
                                  }while(a[i]!='\0');
                                    printf("\nReplaced string:%s",a);
                             }
}
```

21)

```c
#include <stdio.h>

// Recursive function to find the number of times
// the second string occurs in the first string,
// whether continuous or discontinuous
int count(const char* a, const char* b, int m, int n)
{
    // If both first and second string is empty,
    // or if second string is empty, return 1
    if ((m == 0 && n == 0) || n == 0)
        return 1;

    // If only first string is empty and second
    // string is not empty, return 0
    if (m == 0)
        return 0;

    // If last characters are same
    // Recur for remaining strings by
    // 1. considering last characters of both strings
    // 2. ignoring last character of first string
    if (a[m - 1] == b[n - 1])
        return count(a, b, m - 1, n - 1) +
               count(a, b, m - 1, n);
    else
        // If last characters are different, ignore
        // last char of first string and recur for
        // remaining string
        return count(a, b, m - 1, n);
}

// Driver code
int main()
{
    char a[] = "Campus for you";
    char b[] = "cfy";

    printf ("%d",count(a, b, sizeof(a), sizeof(b)));

    return 0;
}
```

22)

```c
#include <stdio.h>

int match(char *first, char * second)
{
    // If we reach at the end of both strings, we are done
    if (*first == '\0' && *second == '\0')
        return 1;

    // Make sure that the characters after '*' are present
    // in second string. This function assumes that the first
    // string will not contain two consecutive '*'
    if (*first == '*' && *(first+1) != '\0' && *second == '\0')
        return 0;

    // If the first string contains '?', or current characters
    // of both strings match
    if (*first == '?' || *first == *second)
        return match(first+1, second+1);

    // If there is *, then there are two possibilities
    // a) We consider current character of second string
    // b) We ignore current character of second string.
    if (*first == '*')
        return match(first+1, second) || match(first, second+1);
    return 0;
}
void test(char *first, char *second)
{
    match(first, second)? puts("Yes"): puts("No");
}

int main()
{
    test("g*ks", "geeks"); // Yes
    test("ge?ks*", "geeksforgeeks"); // Yes
    test("g*k", "gee");  // No because 'k' is not in second
    test("*pqrs", "pqrst"); // No because 't' is not in first
    test("abc*bcd", "abcdhghgbcd"); // Yes
    test("abc*c?d", "abcd"); // No because second must have 2
                             // instances of 'c'
    test("*c*d", "abcd"); // Yes
    test("*?c*d", "abcd"); // Yes
    return 0;
}
```