SAMPLE PROGRAMS WITH STRINGS
“PATTERN MATCHING”

1. Write a function to to find the **position of the first occurrence of a given string** in a sentence if it exists, otherwise returns -1.

```
int findFirstOccSub(char *sent, char *sub)
{
    int k,
        sentlen = strlen(sent),
        wordlen = strlen(sub);
    // Traverse sentence
    for (k = 0; k <= sentlen - wordlen; k++)
    {   /* Compare sub with len2 characters of sent,
        starting from the kth position, to see if they are equal */
        if (strncmp(sent + k, sub, wordlen) == 0)
          return k;
    }
    /* Return -1 if s2 is not found in s1 */
    return -1;
}

//main
position = findFirstOccSub(givensent, givensub);
```

2. Write a function to to find the **position of the first occurrence of a given WORD** in a sentence if it exists, otherwise returns -1.

```
int findFirstOccWord(char *sent, char *word)
{
      int   k, sentlen = strlen(sent),
            wordlen = strlen(word);
      // Traverse sentence
      for (k = 0; k <= sentlen - wordlen; k++)
      {     //Find the first word
            if (strncmp(sent + k, word, wordlen) == 0 &&
                (k == 0 || sent[k - 1] == ' ') &&
                (k == sentlen - wordlen || sent[k + wordlen] == ' '))
                           return k;
      }
      /* Return -1 if word is not found in sent */
      return -1;
}


//main
position = findFirstOccWord(givensent, givenword);
```

3. Write a function to **delete the first occurrence of a given word**

```
// First solution (there will be 2 blank characters)
void deleteFirstOccWord(char *sent, char *word)
{
     int  wordlen = strlen(word),
          pos = findFirstOccWord(sent, word);

     if (pos != -1) {
          char before[SIZE] = "", after[SIZE] = "";
          //take the part of the string BEFORE the given word
          strncpy(before, sent, pos);
          //take the part of the string AFTER the given word
          strcpy(after, sent + pos + wordlen);

          strcpy(sent, before);
          strcat(sent, after);
     }
     else
          printf("The given word does not found !!\n");
}


//main
deleteFirstOccWord(givensent, givenword);
```

```
// Second solution (there will be 2 blank characters)
void deleteFirstOccWord(char *sent, char *word)
{
     int  wordlen = strlen(word),
          pos = findFirstOccWord(sent, word);
     if (pos != -1)
          strcpy(sent + pos, sent + pos + wordlen);
     else
     printf("The given word does not found !!\n");
}

//main
deleteFirstOccWord(givensent, givenword);
```

4. Write a function to **replace the first occurrence of a given word** with the given word.

```
// First solution
void replaceFirstOccWord(char *sent, char *word, char *rep)
{
      int pos = findFirstOccWord(sent, word);

      if (pos != -1) {
            char before[SIZE] = "", after[SIZE] = "";
            int  wordlen = strlen(word);

            //take the part of the string BEFORE the given word
            strncpy(before, sent, pos);

            //insert the given word
            strcat(before, rep);

            //take the part of the string AFTER the given word
            strcpy(after, sent + pos + wordlen);

            strcpy(sent, before);
            strcat(sent, after);
      }
      else
            printf("The given word does not found !!\n");
}
//main
replaceFirstOccWord(givensent, givenword, repwith);
```

```
// Second solution
void replaceFirstOccWord(char *sent, char *word, char *rep)
{
      int   wordlen = strlen(word),
            pos = findFirstOccWord(sent, word);
      if (pos != -1) {
            char temp[SIZE] = "";
            //take the copy of the sent after the given word
            strcpy(temp, sent + pos + wordlen);

            //copy the new word
            strcpy(sent + pos, rep);

            //concatenate the part of the string AFTER the given word
            strcat(sent, temp);
      }
      else
            printf("The given word does not found !!\n");
}

//main
replaceFirstOccWord(givensent, givenword, repwith);
```

5. Write a function to find the **position of all occurrences of a given string** in a sentence if it exists.

```
int findAllSub(char *sent, char *sub, int occ[])
{
     int  k,
          sentlen = strlen(sent),
          sublen = strlen(sub),
          count = 0;
     // Traverse sentence
     for (k = 0; k <= sentlen - sublen; k++)
     {
          /* Compare sub with sublen char. of sent, starting from
          the kth position, to see if they are equal */
          if (strncmp(sent + k, sub, sublen) == 0)
               occ[count++] = k;
     }
     /* Return 0 if sub is not found in sent */
     return count;
}


//main
int posarray[10];
int size = findAllSub(givensent, givensub, posarray);
```

6. Write a function to **delete all occurrences of a given string** in a sentence and finds the new sentence.

```
// First solution
void delAllOcc(char *sent, char *sub)
{
        char before[SIZE] = "", after[SIZE] = "";
        int    sublen = strlen(sub), pos;
        pos = findFirstOccSub(sent, sub);
        while (pos != -1) {
                strncpy(before, sent, pos);
                strcpy(after, sent + pos + sublen);
                strcpy(sent, before);
                strcat(sent, after);
                pos = findFirstOccSub(sent, sub);
        }
}

//main
delAllOcc(givensent, givensub);
```

```
// Second solution
void delAllOcc(char *sent, char *sub)
{
        int    sublen = strlen(sub), pos;
        pos = findFirstOccSub(sent, sub);
        while (pos != -1) {
                strcpy(sent + pos, sent + pos + sublen);
                pos = findFirstOccSub(sent, sub);
        }
}

//main
delAllOcc(givensent, givensub);
```