String Functions Assignment

1. WAP to prompt and read a line of text with words separated by space. Tokenise and extract the words. Display them. Impement the following functions for this.

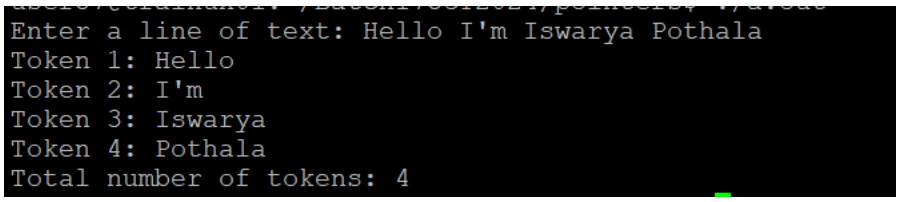
a. int tokenise(char *arr); //tokenise and display tokens , return number of tokens to the caller

[Use strtok() to tokenise]

sol:

```
#include <stdio.h>

#include <string.h>

#define MAX 100

// Function to tokenize the string and display each token

int tokenise(char *arr) {

    int count = 0;

    char *token;

 // Get the first token

    token = strtok(arr, " ");

    while (token != NULL) {

        printf("Token %d: %s\n", ++count, token);

        token = strtok(NULL, " ");

    }

   return count;

}

int main() {

    char input[MAX];

    printf("Enter a line of text: ");

    fgets(input, sizeof(input), stdin);  // Read the input

   // Remove the trailing newline character if any

    input[strcspn(input, "\n")] = '\0';


    // Call the tokenise function to get tokens and display them
    int numTokens = tokenise(input);

   // Display the number of tokens

    printf("Total number of tokens: %d\n", numTokens);

   return 0;

}
```

Output:



```
Enter a line of text: Hello I'm Iswarya Pothala
Token 1: Hello
Token 2: I'm
Token 3: Iswarya
Token 4: Pothala
Total number of tokens: 4
```

2. Implement your own strncat() which shall concatenate n characters from src to dest.

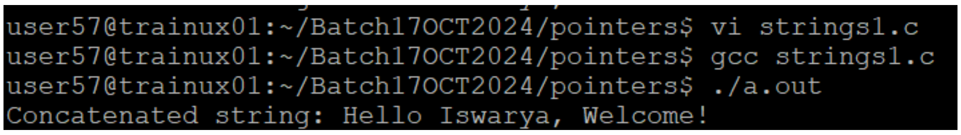
char *strncat(char *dest, const char *src, size_t n)

sol:

```
#include <stdio.h>

#define MAX 100

char *my_strncat(char *dest, const char *src, size_t n) {

    while (*dest != '\0') {

        dest++;

    }

   // Copy n characters from src to dest

    size_t count = 0;

    while (count < n && *src != '\0') {

        *dest = *src;  // Copy the character

        dest++;       // Move to the next position in dest

        src++;       // Move to the next character in src

        count++;      // Increment the count

    }

   *dest = '\0';

    return dest;

}
```

```
int main() {
    char dest[MAX] = "Hello, ";
    char src[] = "world!";
  // Concatenate first 10 characters of src to dest
    my_strncat(dest, src, 10);
    printf("Concatenated string: %s\n", dest);


    return 0;
}
```

Output:



```
user57@trainux01:~/Batch17OCT2024/pointers$ vi strings1.c
user57@trainux01:~/Batch17OCT2024/pointers$ gcc strings1.c
user57@trainux01:~/Batch17OCT2024/pointers$ ./a.out
Concatenated string: Hello Iswarya, Welcome!
```

3. WAP to

a. Search for and replace the vowel characters (upper and lower case) with "ay" in a word read from user. Consider a maximum word length of 40 characters.

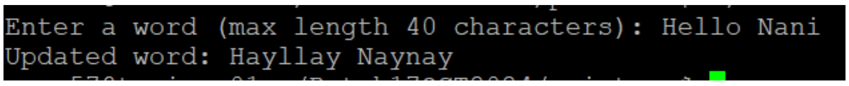
b. Identify the test inputs to be used

c. Display updated string

sol:

```
#include <stdio.h>
#include <string.h>
#define MAX_LEN 40
// Function to replace vowels with "ay"
void replace_vowels_with_ay(char *str) {
    char result[MAX_LEN * 2];     // To store the updated string (since "ay" is 2 chars for each vowel)
    int j = 0;


    for (int i = 0; str[i] != '\0'; i++) {
        // Check if the character is a vowel
        if (str[i] == 'a' || str[i] == 'A' || str[i] == 'e' || str[i] == 'E' ||
          str[i] == 'i' || str[i] == 'I' || str[i] == 'o' || str[i] == 'O' ||
          str[i] == 'u' || str[i] == 'U') {
          // If vowel, add "ay" to result
          result[j++] = 'a';
          result[j++] = 'y';
        } else {
          result[j++] = str[i];
        }
    }
    result[j] = '\0';
    strcpy(str, result);  // Copy the result back to the original string
}

int main() {
    char word[MAX_LEN];
    printf("Enter a word (max length 40 characters): ");
    fgets(word, sizeof(word), stdin);
    word[strcspn(word, "\n")] = '\0';   //trailing the new line

    replace_vowels_with_ay(word);
    printf("Updated word: %s\n", word);
   return 0;
}
```

Output:



```
Enter a word (max length 40 characters): Hello Nani
Updated word: Hayllay Naynay
```