LAB ASSIGNMENT 14

1. Given a string “str” of size “n”. Check whether it’s a palindrome or not.

```
#include <stdio.h>


// Function to check if the string is palindrome
// using pointers
void isPalindrome(char* string)
{
   char *ptr, *rev;

   ptr = string;

   while (*ptr != '\0') {
      ++ptr;
   }
   --ptr;

   for (rev = string; ptr >= rev;) {
      if (*ptr == *rev) {
         --ptr;
         rev++;
      }
      else
```

Samuela Abigail
71762108039

```
            break;
    }

    if (rev > ptr)
        printf("String is Palindrome");
    else
        printf("String is not a Palindrome");
}

// Driver code
int main()
{
    char str[1000] = "madam";

    isPalindrome(str);

    return 0;
}
```

2. Copy a source string into destination string.

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

void copy_string(char*, char*);
```

```
main()
{
   char source[100], target[100];
   printf("Enter source string\n");
   scanf("%s",source);
   copy_string(target, source);
   printf("Target string is \"%s\"\n", target);
   return 0;
}

void copy_string(char *target, char *source)
{
   while(*source)
   {
      *target = *source;
      source++;
      target++;
   }
   *target = '\0';
}
```

Samuela Abigail
71762108039

3. Insert a string “J” into string “I” at position “pos”.

```
#include <string.h>
#include <stdlib.h>
void insert_substring(char*, char*, int);
char* substring(char*, int, int);

int main()
{
  char text[100], substring[100];
  int position;

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

  printf("Enter a string to insert\n");
  scanf("%s",substring);

  printf("Enter the position to insert\n");
  scanf("%d", &position);

  insert_substring(text, substring, position);

  printf("%s\n",text);
```

Samuela Abigail
71762108039

```
  return 0;
}

void insert_substring(char *a, char *b, int position)
{
  char *f, *e;
  int length;

  length = strlen(a);

  f = substring(a, 1, position - 1 );
  e = substring(a, position, length-position+1);

  strcpy(a, "");
  strcat(a, f);
  free(f);
  strcat(a, b);
  strcat(a, e);
  free(e);
}

char *substring(char *string, int position, int length)
{
  char *pointer;
```

```
  int c;

  pointer = malloc(length+1);

  if( pointer == NULL )
     exit(EXIT_FAILURE);

  for( c = 0 ; c < length ; c++ )
    *(pointer+c) = *((string+position-1)+c);

  *(pointer+c) = '\0';
  return pointer;
}
```

Samuela Abigail
71762108039