ASSIGNMENT:8 RITU KUMARI 115001230105

1. Write a C function to swap two numbers using pointers.
code:

```
#include <stdio.h>
void swap(int *a, int *b) {
   int temp = *a;
   *a = *b;
   *b = temp;
}

int main() {
   int a1 = 5, a2 = 10;

   printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

   swap(&a1, &a2);

   printf("After swapping: a1 = %d, a2 = %d\n", a1, a2);

   return 0;
}
```

OUTPUT:

```
(base) linuxmint@linuxmint-HP-ProDesk-400-G4-MT:~/prachipatil_088$ ./a.out
 enter two numbers:15
9
 before swap: a1=15, a2=9
after swap:a1=9,a2=15
```

2.WACP which will read a text and count all occurrence of a particular word.

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

int main() {
   int n = 100;
   char str[n];

   printf("Input String: ");
   fgets(str, sizeof(str), stdin);

   char letter;
   printf("Input letter to be searched: ");
   scanf("%c", &letter);

   char *ptr = str;
   int word_cnt = 0;

   while (*ptr != '\0') {
      if (*ptr == ' ' || *ptr == '\n') {
         ptr++;
         continue; // Skip spaces and newline characters
      }

      int cnt = 0; // Counter for the current word
      char *word_ptr = ptr;
```

```
    while (*ptr != ' ' && *ptr != '\n' && *ptr != '\0') {
      if (*ptr == letter) {
        cnt++;
      }
      ptr++;
    }

    if (cnt > 0) {
      word_cnt++;
      printf("The value '%c' has occurred %d time(s) in word %d.\n", letter, cnt, word_cnt);
    }

    // Move to the next word
    ptr = word_ptr;
    while (*ptr != ' ' && *ptr != '\n' && *ptr != '\0') {
      ptr++;
    }
  }

  return 0;
}
```

OUTPUT:

```
(base) linuxmint@linuxmint-HP-ProDesk-400-G4-MT:~/prachipatil_088$ ./a
Input String: MY NAME IS PRACHI
Input letter to be searched: A
The value 'A' has occurred 1 time(s) in word 1.
The value 'A' has occurred 1 time(s) in word 2.
```

3. Write a C function to find the largest of three numbers using pass by reference. The function to find the largest should return a pointer.

Solution:

```
#include <stdio.h>

int* findLargest(int* a, int* b, int* c) {
  if (*a >= *b && *a >= *c)
    return a;
  else if (*b >= *a && *b >= *c)
    return b;
  else
    return c;
}

int main() {
  int num1, num2, num3;
  printf("Enter three numbers: ");
  scanf("%d %d %d", &num1, &num2, &num3);

  int* largest = findLargest(&num1, &num2, &num3);

  printf("The largest number is: %d\n", *largest);

  return 0;
}
```

OUTPUT:

```
(base) linuxmint@linuxmint-HP-ProDesk-400-G4-MT:~/prachipatil_088$ ./a.out
Enter three number:12 9 10
The largest number is:12
```

4. WACP using pointers to read in an array of integers and print its elements in reverse order.

```
#include <stdio.h>
void printReverse(int *arr, int size) {
   int *ptr = arr + size - 1;
   printf("Elements in reverse order:\n");
   for (; ptr >= arr; ptr--) {
      printf("%d ", *ptr);
   }
   printf("\n");
}
int main() {
   int size;
   printf("Enter the size of the array: ");
   scanf("%d", &size);
   int arr[size];
   printf("Enter %d integers:\n", size);
   for (int i = 0; i < size; i++) {
      scanf("%d", &arr[i]);
   }
   printReverse(arr, size);
   return 0;
}
```

OUTPUT:

```
Enter the size of the array: 5
Enter 5 integers:
12
30
13
24
22
Elements in reverse order:
22 24 13 30 12
```

5.WACP to sort strings abc, deh, bac in ascending order using array of pointers.

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

#define NUM_STRINGS 3

void sortStrings(char *arr[], int n) {
   int i, j;
   char *temp;
```

```
    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (strcmp(arr[i], arr[j]) > 0) {
                // Swap pointers
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
}

int main() {
    char *strings[NUM_STRINGS] = {"abc", "deh", "bac"};
    int i;

    printf("Before sorting:\n");
    for (i = 0; i < NUM_STRINGS; i++) {
        printf("%s\n", strings[i]);
    }

    sortStrings(strings, NUM_STRINGS);

    printf("\nAfter sorting:\n");
    for (i = 0; i < NUM_STRINGS; i++) {
        printf("%s\n", strings[i]);
   for (i = 0; i < NUM_STRINGS; i++) {
        printf("%s\n", strings[i]);
    }

    return 0;
}
}
```

OUTPUT: