**NITHIN S**
**221IT085**

# CS111 Lab Assignment 7

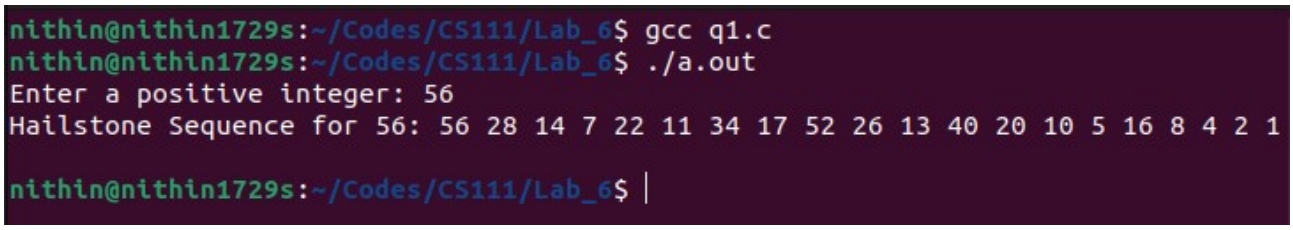
# Recursion and Pointers

**Q1) Write a program in C to find the Hailstone Sequence of a given number up to 1 using recursion.**

```
#include <stdio.h>


void hailstoneSequence(int n) {
    printf("%d ", n);

    if (n == 1) {
        return;
    } else if (n % 2 == 0) {
        hailstoneSequence(n / 2);
    } else {
        hailstoneSequence((n * 3) + 1);
    }
}

int main() {
    int number;

    printf("Enter a positive integer: ");
    scanf("%d", &number);

    if (number <= 0) {
        printf("Please enter a positive integer.\n");
    } else {
        printf("Hailstone Sequence for %d: ", number);
        hailstoneSequence(number);
```

```
        printf("\n");
    }

    return 0;
}
```

## OUTPUT

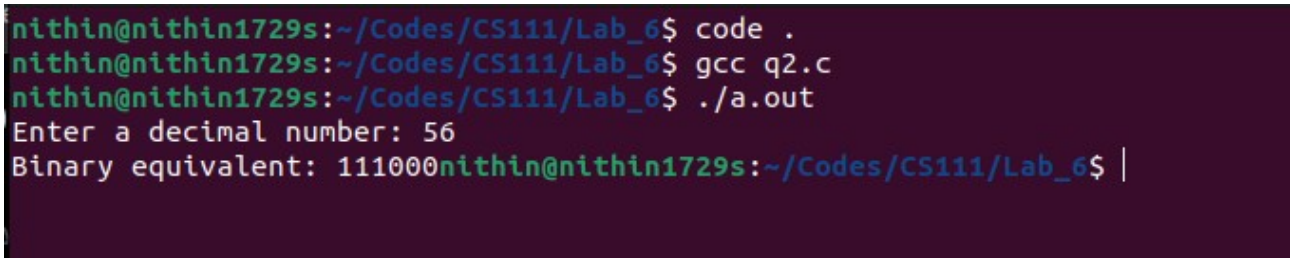



**Q2) Write a C program to convert decimal number to binary using recursion with appropriate parameter passing**

```
#include <stdio.h>

void decimalToBinary(int n) {
    if (n > 1) {
        decimalToBinary(n / 2);
    }
    printf("%d", n % 2);
}

int main() {
    int decimalNumber;

    printf("Enter a decimal number: ");
    scanf("%d", &decimalNumber);

    printf("Binary equivalent: ");
    decimalToBinary(decimalNumber);
```

```
    return 0;
}
```

## OUTPUT




**Q3) Write a C program to perform bubble sort on an integer array using recursion with appropriate parameter passing**

```
#include <stdio.h>

void bubbleSort(int arr[], int size) {
    if (size == 1) {
        return;
    }


    for (int i = 0; i < size - 1; i++) {
        if (arr[i] > arr[i + 1]) {

            int temp = arr[i];
            arr[i] = arr[i + 1];
            arr[i + 1] = temp;
        }
    }


    bubbleSort(arr, size - 1);
}

int main() {
```

```
    int size;
    printf("Enter the size of the array: ");
    scanf("%d", &size);

    int arr[size];

    printf("Enter %d elements for the array: ", size);
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Original array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }

    bubbleSort(arr, size);

    printf("\nSorted array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

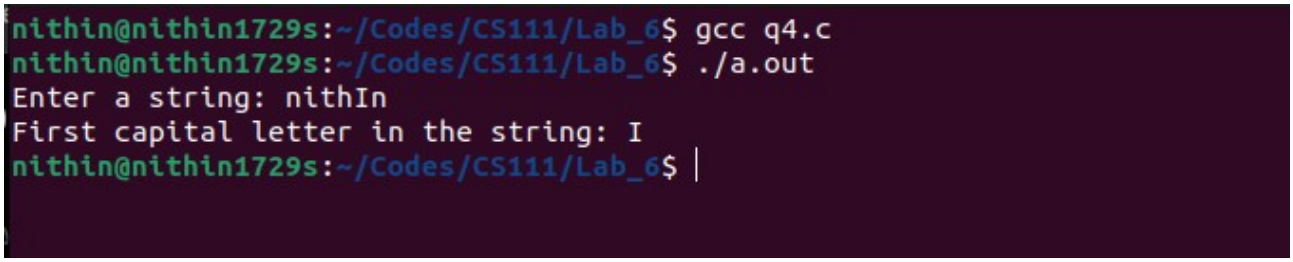
## OUTPUT


### Q4) Write a C program to find the maximum occurring character in a string.

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

```
char firstCapitalLetter(char str[], int index) {
    if (str[index] == '\0') {
        return '\0';
    }

    if (str[index] >= 'A' && str[index] <= 'Z') {
        return str[index];
    }

    return firstCapitalLetter(str, index + 1);
}

int main() {
    char inputString[100];

    printf("Enter a string: ");
    scanf("%s", inputString);

    char result = firstCapitalLetter(inputString, 0);

    if (result != '\0') {
        printf("First capital letter in the string: %c\
n", result);
    } else {
        printf("No capital letter found in the string.\
n");
    }

    return 0;
}
```

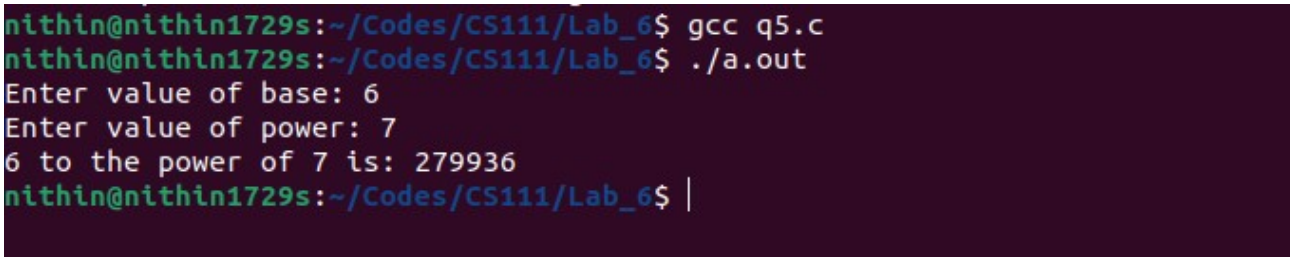
## OUTPUT



**Q5)  C program to calculate power of a number using recursion**

```
#include <stdio.h>


int power(int base, int exponent) {
    if (exponent == 0) {
        return 1;
    } else {
        return base * power(base, exponent - 1);
    }
}

int main() {
    int base, exponent;

    printf("Enter value of base: ");
    scanf("%d", &base);

    printf("Enter value of power: ");
    scanf("%d", &exponent);

    int result = power(base, exponent);

    printf("%d to the power of %d is: %d\n", base,
exponent, result);

    return 0;
}
```

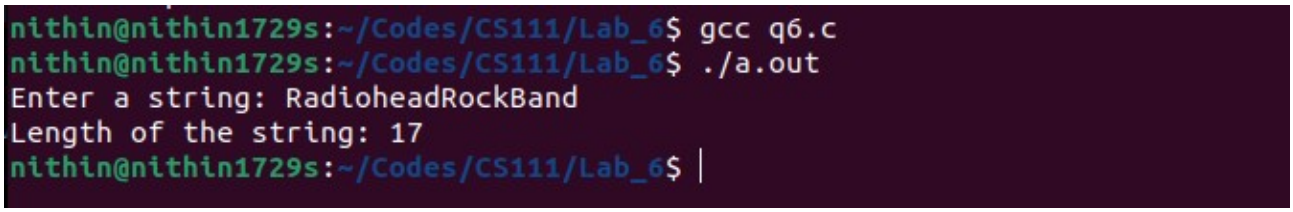
## OUTPUT



**Q6) Write a C program to find length of the string using pointers and functions, showing appropriate parameter passing.**

```
#include <stdio.h>


int stringLength(const char *str) {
    int length = 0;

    while (*str != '\0') {
        length++;
        str++;
    }

    return length;
}

int main() {
    char inputString[100];

    printf("Enter a string: ");
    scanf("%s", inputString);


    int len = stringLength(inputString);

    printf("Length of the string: %d\n", len);

    return 0;
}
```

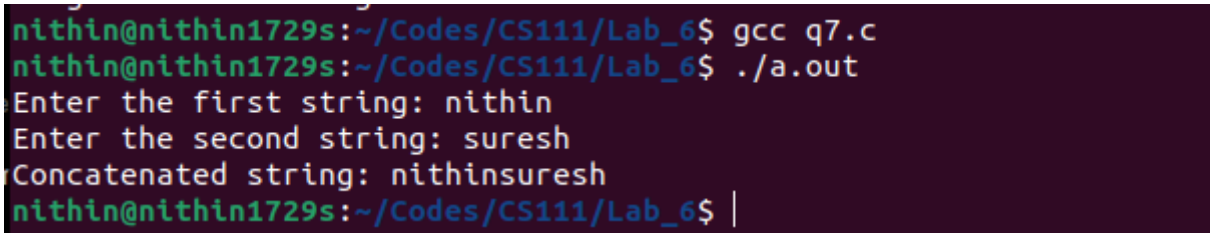
## OUTPUT



**Q7) Write a C program to concatenate two strings using pointers and functions, showing appropriate parameter passing**

```
#include <stdio.h>


void stringConcat(char *dest, const char *src) {

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


    while (*src != '\0') {
        *dest = *src;
        dest++;
        src++;
    }


    *dest = '\0';
}

int main() {
    char firstString[100], secondString[100];

    printf("Enter the first string: ");
    scanf("%s", firstString);

    printf("Enter the second string: ");
    scanf("%s", secondString);


    stringConcat(firstString, secondString);

    printf("Concatenated string: %s\n", firstString);

    return 0;
}
```

## OUTPUT

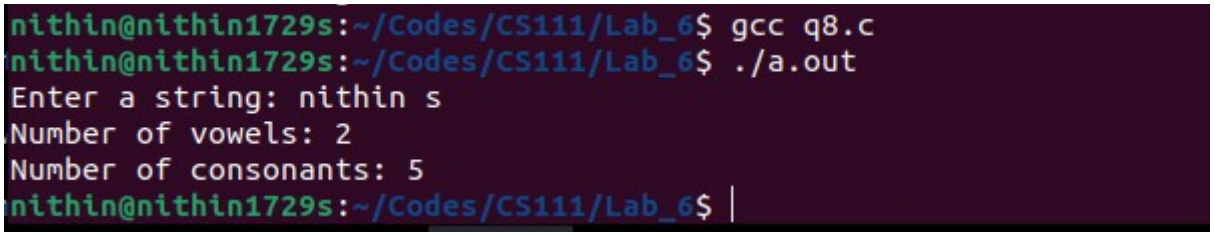



## Q8) Write a program in C to count the number of vowels and consonants in a string using a pointer

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


void countVowelsConsonants(const char *str, int
*vowelCount, int *consonantCount) {
    *vowelCount = 0;
    *consonantCount = 0;

    while (*str != '\0') {
        char ch = toupper(*str);

        if (ch == 'A' || ch == 'E' || ch == 'I' || ch ==
'O' || ch == 'U') {
            (*vowelCount)++;
        } else if (ch >= 'A' && ch <= 'Z') {
            (*consonantCount)++;
        }

        str++;
    }
}

int main() {
    char inputString[100];
    int vowelCount = 0, consonantCount = 0;

    printf("Enter a string: ");
    scanf("%[^\n]", inputString);
```

```
    countVowelsConsonants(inputString, &vowelCount,
&consonantCount);

    printf("Number of vowels: %d\n", vowelCount);
    printf("Number of consonants: %d\n", consonantCount);

    return 0;
}
```

## OUTPUT

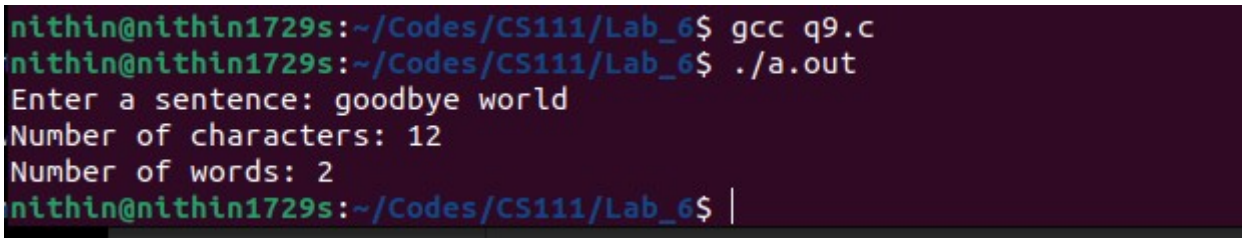



**Q9)Write a C program to read a sentence and count the number of characters and words in that sentence, using pointers and functions, showing appropriate parameter passing.**

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


void countCharsWords(const char *sentence, int
*charCount, int *wordCount) {
    *charCount = 0;
    *wordCount = 0;
    int inWord = 0;

    while (*sentence != '\0') {
        if (!isspace(*sentence)) {
            (*charCount)++;

            if (!inWord) {
                inWord = 1;
                (*wordCount)++;
```

```
            }
        } else {
            inWord = 0;
        }

        sentence++;
    }
}

int main() {
    char inputSentence[1000];
    int charCount = 0, wordCount = 0;

    printf("Enter a sentence: ");
    scanf("%[^\n]", inputSentence);


    countCharsWords(inputSentence, &charCount,
&wordCount);

    printf("Number of characters: %d\n", charCount);
    printf("Number of words: %d\n", wordCount);

    return 0;
}
```

## OUTPUT

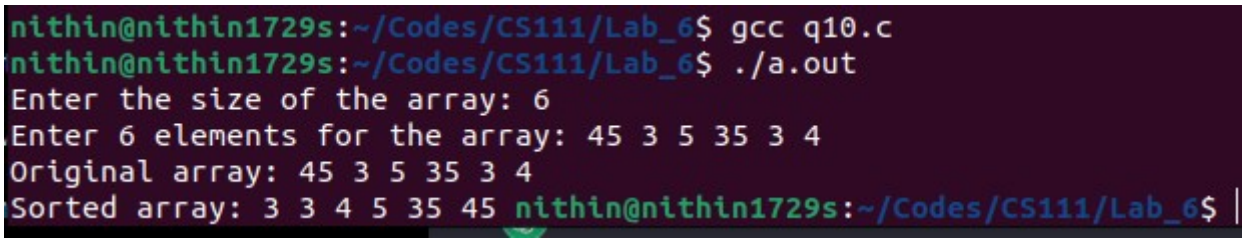



### Q10) Write a C program to sort array elements using pointers.

```
#include <stdio.h>


void swap(int *a, int *b) {
    int temp = *a;
```

```
    *a = *b;
    *b = temp;
}


void sortArray(int *arr, int size) {
    for (int i = 0; i < size - 1; i++) {
        for (int j = 0; j < size - i - 1; j++) {
            if (*(arr + j) > *(arr + j + 1)) {
                swap(arr + j, arr + j + 1);
            }
        }
    }
}

int main() {
    int size;

    printf("Enter the size of the array: ");
    scanf("%d", &size);

    int arr[size];

    printf("Enter %d elements for the array: ", size);
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Original array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }


    sortArray(arr, size);

    printf("\nSorted array: ");
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }

    return 0;
}
```

## OUTPUT


nithin@nithin1729s:~/Codes/CS111/Lab_6$ gcc q10.c
nithin@nithin1729s:~/Codes/CS111/Lab_6$ ./a.out
Enter the size of the array: 6
Enter 6 elements for the array: 45 3 5 35 3 4
Original array: 45 3 5 35 3 4
Sorted array: 3 3 4 5 35 45 nithin@nithin1729s:~/Codes/CS111/Lab_6$