Name: Sneha Roy , Section : B , Roll : 42

Enrollment No.: 12024006015099

Assignment - 2

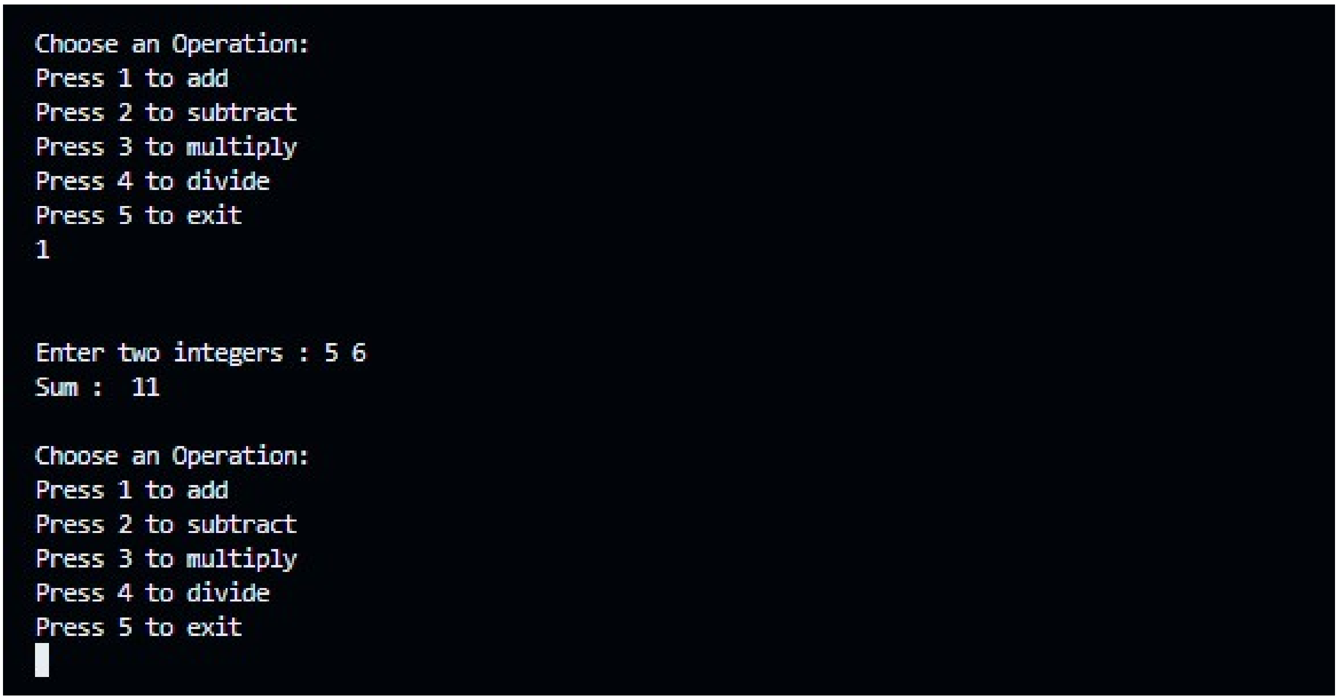
1. Write a C program to add, subtract, multiply, and divide two integers using a user-defined type function with a return type.

Input:

```
#include <stdio.h>

int add (int a, int b){

  return a + b;

}

int subtract (int a, int b){

  return a - b;

}

int multiply (int a, int b){

  return a * b;

}

int divide (int a, int b){

  return (float)a / b;

}
```

```
int main() {
 int a, b, choice;
 do {
  printf("\n\nChoose an Operation: \n");
  printf("Press 1 to add\n");
  printf("Press 2 to subtract\n");
  printf("Press 3 to multiply\n");
  printf("Press 4 to divide\n");
  printf("Press 5 to exit\n");
  scanf("%d", &choice);
  printf("\n\nEnter two integers : ");
  scanf("%d %d", &a, &b);
  switch (choice){
  case 1:
   printf("Sum :  %d", add(a,b));
   break;
  case 2:
   printf("Difference :  %d", subtract(a,b));
   break;
  case 3:
   printf("Product :  %d", multiply(a,b));
   break;
   case 4:
     if(b != 0)printf("Quotient :  %d", divide(a,b));
     else printf("Error: Division by Zero!");
     break;
   case 5:
     printf("Exit");
     break;
   default: printf("Invalid Choice!!");
   }
 } while (choice != 5);
 return 0;
}
```

Output:



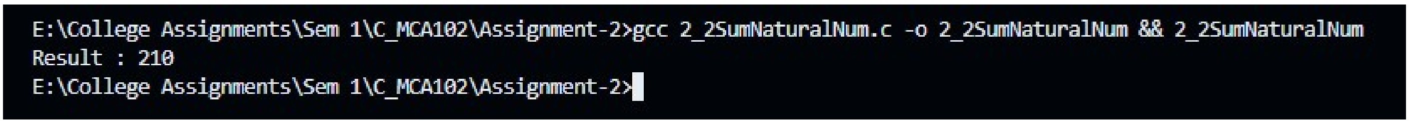
```
Choose an Operation:
Press 1 to add
Press 2 to subtract
Press 3 to multiply
Press 4 to divide
Press 5 to exit
1


Enter two integers : 5 6
Sum :  11

Choose an Operation:
Press 1 to add
Press 2 to subtract
Press 3 to multiply
Press 4 to divide
Press 5 to exit
```

2. Write a C program to calculate the sum of the first 20 natural numbers using a recursive function.

Input:

```
#include<stdio.h>

int sum(int n){

  if(n == 0 || n == 1) return n;

 else  return n + sum(n - 1);

}

int main(){

  printf("Result : %d", sum(20));

  return 0;

}
```

Output:



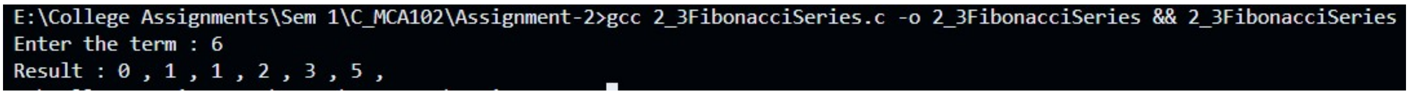
```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_2SumNaturalNum.c -o 2_2SumNaturalNum && 2_2SumNaturalNum
Result : 210
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>
```

3. Write a C program to generate a Fibonacci series using a recursive function.

Input:

```
#include<stdio.h>

int Fibonacci(int term){

  if(term == 0 || term == 1) return term;

  else return Fibonacci(term - 1) + Fibonacci(term - 2);

}

int main(){

  int term;

  printf("Enter the term : ");

  scanf("%d", &term);

  printf("Result : " );

  for(int i = 0; i < term; i++) printf("%d , ", Fibonacci(i));

  return 0;

}
```

Output:



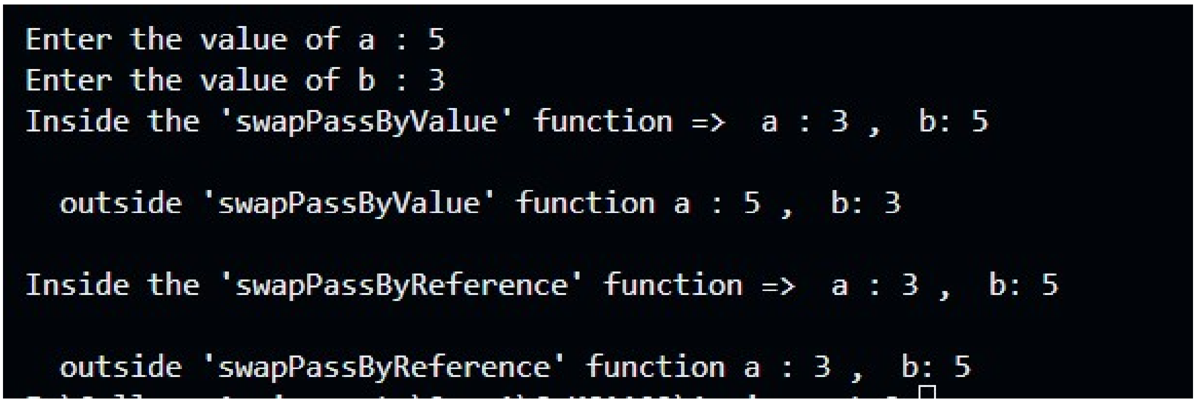
```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_3FibonacciSeries.c -o 2_3FibonacciSeries && 2_3FibonacciSeries
Enter the term : 6
Result : 0 , 1 , 1 , 2 , 3 , 5 ,
```

4. Write a C program to swap two integers using call-by-value and call-by-reference methods of passing arguments to a function.

Input:

```
#include<stdio.h>

void swapPassByValue(int a, int b){

  int temp = a;

  a = b;

  b = temp;

  printf("Inside the 'swapPassByValue' function =>  a : %d ,  b: %d \n" , a ,b);

}

void swapPassByReference(int *a, int *b){

  int temp = *a;

  *a = *b;

  *b = temp;

  printf("\n\nInside the 'swapPassByReference' function =>  a : %d ,  b: %d \n" , *a ,*b);

}


int main(){

  int a, b;

  printf("Enter the value of a : ");

  scanf("%d", &a);

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

  //Pass By value

  swapPassByValue(a, b);

  printf("\n  outside 'swapPassByValue' function a : %d ,  b: %d " , a ,b);  //no change

  //Pass by Reference

  swapPassByReference(&a, &b);

  printf("\n  outside 'swapPassByReference' function a : %d ,  b: %d " , a ,b);  //no change

  return 0;

}
```

Output:



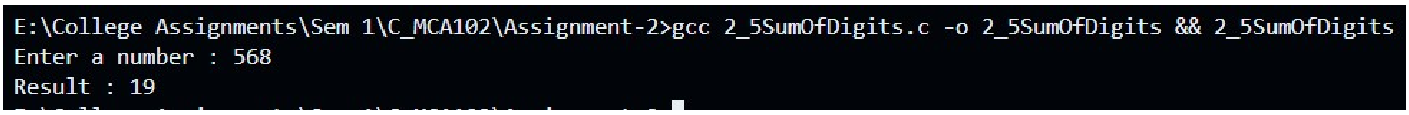
```
Enter the value of a : 5
Enter the value of b : 3
Inside the 'swapPassByValue' function =>  a : 3 ,  b: 5

  outside 'swapPassByValue' function a : 5 ,  b: 3

Inside the 'swapPassByReference' function =>  a : 3 ,  b: 5

  outside 'swapPassByReference' function a : 3 ,  b: 5
```

5. Write a C program to find the sum of the digits of the number using a recursive function.

Input:

```
#include<stdio.h>

int sumOfDigit(int n){

  if(n == 0) return 0;

 else  return (n% 10 ) + sumOfDigit(n / 10);

}


int main(){

  int n;

  printf("Enter a number : ");

  scanf("%d", &n);

  printf("Result : %d", sumOfDigit(n));

  return 0;

}
```

Output:



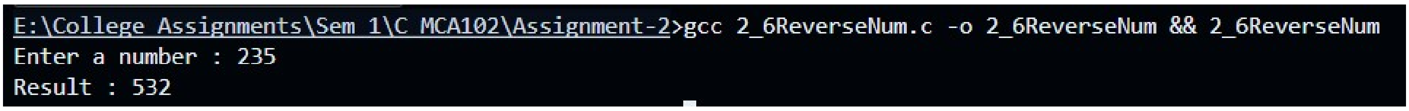
```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_5SumOfDigits.c -o 2_5SumOfDigits && 2_5SumOfDigits
Enter a number : 568
Result : 19
```

6. Write a C program to read an integer number and print the reverse of that number using recursion.

Input:

```
#include<stdio.h>

void reverse(int n){

  if(n == 0) return ;

  else {

    printf("%d", n%10);

    return reverse(n / 10);

  }

}

int main(){

  int n;

  printf("Enter a number : ");

  scanf("%d", &n);

  printf("Result : ");

  reverse(n);

  return 0;

}
```

Output:



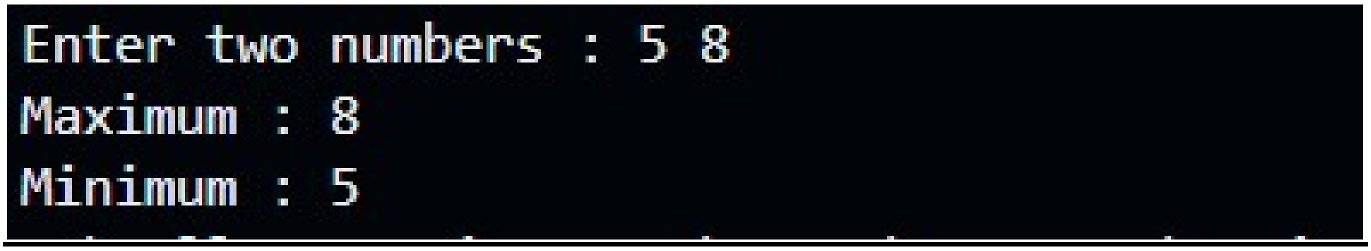
```
E:\College Assignments\Sem 1\C MCA102\Assignment-2>gcc 2_6ReverseNum.c -o 2_6ReverseNum && 2_6ReverseNum
Enter a number : 235
Result : 532
```

7. Using functions, write a C program to find the maximum and minimum between two numbers.

Input:

```
#include<stdio.h>
int max(int a, int b){
 if(a > b) return a;
 else return b;
}
int min(int a, int b){
 if(a < b) return a;
 else return b;
}
int main(){
  int a , b;
  printf("Enter two numbers : ");
  scanf("%d %d", &a, &b);
  printf("Maximum : %d\n", max(a,b));
  printf("Minimum : %d", min(a,b));
  return 0;
}
```

Output:



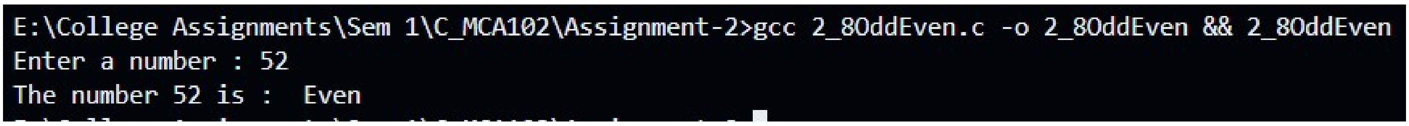
```
Enter two numbers : 5 8
Maximum : 8
Minimum : 5
```

8. Write a C program to check whether a number is even or odd using functions.

Input:

```
#include<stdio.h>
char* OddEven(int n){
  if(n == 0) return "Zero";
  else if( n % 2 == 0 ) return "Even";
  else return "Odd";
}
int main(){
  int num;
  printf("Enter a number : ");
  scanf("%d", &num);

  printf("The number %d is :  %s", num, OddEven(num));
  return 0;
}
```

Output:



```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_8OddEven.c -o 2_8OddEven && 2_8OddEven
Enter a number : 52
The number 52 is :  Even
```

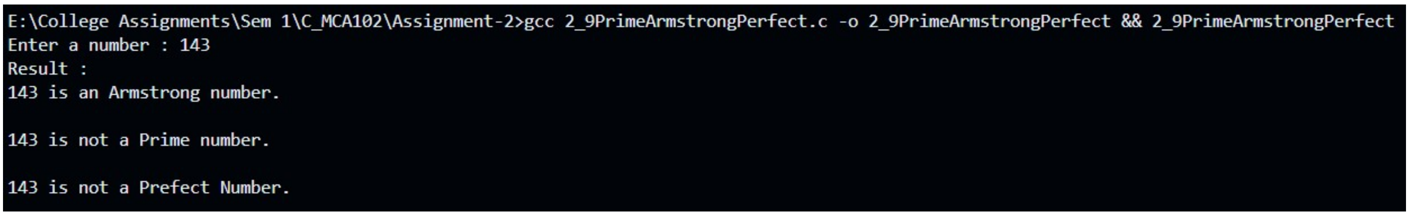
9. Write a C program to check whether a number is a prime, Armstrong, or Perfect number using functions.

Input:

```
#include<stdio.h>

void isArmstrong (int num){

 int n, digitCount = 0, remainder, sum = 0;

 n = num;

  while (n != 0) {

    n = n / 10;

    digitCount++;

  }

  n = num;

  while (n != 0) {

    remainder = n % 10;

    int power = 1;

    for (int i = 0; i < digitCount; i++) {

      power *= remainder;

    }

    sum += power;

    n /= 10;

  }

  if( num == sum) printf("%d is an Armstrong number.\n\n", num);
```

```
    else printf("%d is an Armstrong number.\n\n", num);
}
void isPrime (int n){
  for (int i = 2; i < n; i++) {
    if(n % i == 0) {
      printf("%d is not a Prime number.\n\n" , n);
      return;
    }
  }
  if(n == 1) printf("1 is not a Prime Number.\n\n");
  else printf("%d is a Prime Number.\n\n", n);
}

void isPrefect(int n){
  int sum = 0;

  for(int i = 1; i < n; i++){
    if(n % i == 0){
      sum += i;
    }
  }
  if(sum == n) printf("%d is a Prefect Number.\n\n", n);
  else printf("%d is not a Prefect Number.\n\n", n);


}


int main(){
  int n;
  printf("Enter a number : ");
  scanf("%d", &n);
  printf("Result : \n");
  isArmstrong(n);
  isPrime(n);
  isPrefect(n);
  return 0;
}
```

Output:



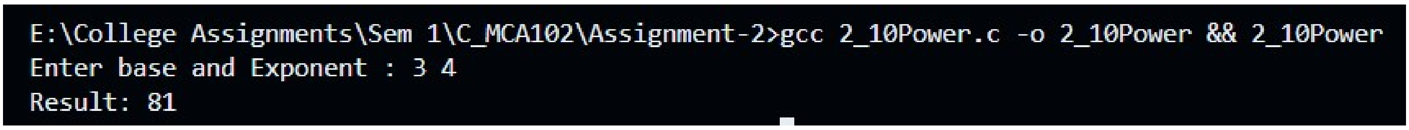
```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_9PrimeArmstrongPerfect.c -o 2_9PrimeArmstrongPerfect && 2_9PrimeArmstrongPerfect
Enter a number : 143
Result :
143 is an Armstrong number.

143 is not a Prime number.

143 is not a Prefect Number.
```

10. Write a C program to find the power of any number using recursion.

Input:

```
#include<stdio.h>
int power(int base, int pow){
  if(pow == 1) return base;
  return base * power(base, pow - 1);
}
int main(){
  int base, pow;
  printf ("Enter base and Exponent : ");
  scanf("%d %d", &base, &pow);

  printf("Result: %d", power(base, pow));
  return 0;
}
```

Output:



```
E:\College Assignments\Sem 1\C_MCA102\Assignment-2>gcc 2_10Power.c -o 2_10Power && 2_10Power
Enter base and Exponent : 3 4
Result: 81
```