**SSN COLLEGE OF ENGINEERING (Autonomous)**

**Affiliated to Anna University**

**DEPARTMENT OF CSE**

**UCS 1211 PROGRAMMING IN C LABORATORY**

**A2 : Modular Programming with C**

========================================================

**REGISTRATION NO: 185001121**

**NAME: RAHUL RAM M**

**CLASS: CSE-B (SEMESTER-2)**

========================================================

***1. Modify A1(1) to have a function CheckOddEven(num) that checks if the num is odd or even; sets a flag accordingly and return it. Use this function to find the sum of even and odd numbers in a given input of N numbers.***

**PROGRAM:**

#include<stdio.h>

void main()

{

int flag,n,num,i,odd=0,even=0;

printf("Enter the number of terms:");

scanf("%d",&n);

for(i=0;i<n;i++)

{

printf("Enter the %dth number:",i+1);

scanf("%d",&num);

flag=CheckOddEven(num);

if(flag==1)

{

even=even+num;

}

else

{

odd=odd+num;

}

}

printf("The sum of even number is :%d\n",even);

printf("The sum of odd number is :%d\n",odd);

}

int CheckOddEven(int num)

{

if(num%2==0)

{

flag=1;

}

else

{

flag=0;

}

Retrun flag;

}

**Output:**

cseb121@jtl-19:~$gcc funoddeven.c –o a2\_1

cseb121@jtl-19:~$./ a2\_1

Enter the number of terms:5

Enter the 1th number:1

Enter the 2th number:2

Enter the 3th number:3

Enter the 4th number:4

Enter the 5th number:5

The sum of even number is :6

The sum of odd number is :9

***2. Write a C function ReverseNum(num) that takes integer num and reverses its digits. Let num be passed by reference.***

**PROGRAM:**

#include<stdio.h>

void reversenum(int \*num);

int main()

{

int num;

printf("Enter a number:");

scanf("%d",&num);

reversenum(&num);

return 0;

}

void reversenum(int \*num)

{

int ans=0;

while(\*num>0)

{

ans\*=10;

ans=ans+\*num%10;

\*num=\*num/10;

}

printf("The answer is:%d\n",ans);

}

**Output:**

cseb121@jtl-19:~$gcc reversenum.c –o a2\_2

cseb121@jtl-19:~$./ a2\_2

Enter a number:35656567

The answer is:76565653

***3. Write a function power(X,N) that will allow a floating-point number to be raised to an integer power. Y = X N In other words, evaluate the formula where y and x are floating-point variables and n is an integer variable. Write a C program that will read in numerical values for x and n, evaluate the formula using power(X,N), and then display the calculated result.***

**PROGRAM:**

#include<stdio.h>

float power(float x,int n)

{

float y=1;

if (n>0)

{

for(int i=0;i<n;i++)

{

y=y\*x;

}

return y;

}

else if(n==0)

{

return 1;

}

else

{

for(int i=0;i>n;i--)

{

y=y\*x;

}

return 1/y;

}}

int main()

{

int n;

float x,result;

printf("Enter the number and its power to be raised:");

scanf("%f %d",&x,&n);

result=power(x,n);

printf("Answer is:%f\n",result);

return 0;

}

**Output:**

cseb121@jtl-19:~$gcc power.c –o a2\_3

cseb121@jtl-19:~$./ a2\_3

Enter the number and its power to be raised:2 3

Answer is:8.000000

cseb121@jtl-19:~$gcc power.c –o a2\_3

cseb121@jtl-19:~$./ a2\_3

Enter the number and its power to be raised:-3 -5

Answer is:-0.004115

cseb121@jtl-19:~$gcc power.c –o a2\_3

cseb121@jtl-19:~$./ a2\_3

Enter the number and its power to be raised:0.2 -5

Answer is:3124.999756

***4. Find the product of n floating point numbers. The numbers should be read from the keyboard. You should not use any looping construct. [Hint: use recursion and decide a suitable sentinel for termination of recursion.]***

**PROGRAM:**

#include<stdio.h>

void main()

{

int n;

float ans;

float productrec(int n);

printf("Enter the number of terms:");

scanf("%d",&n);

ans=productrec(n);

printf("The answer is:%f\n",ans);

}

float productrec(int n)

{

float count;

if(n==0)

{

return 1;

}

else

{

printf("enter the number:");

scanf("%f",&count);

return(count\*productrec(n-1));

}

}

**Output:**

cseb121@jtl-19:~$gcc productrec.c –o a2\_4

cseb121@jtl-19:~$./ a2\_4

Enter the number of terms:4

enter the number:3.5

enter the number:3.5

enter the number:6.5

enter the number:7.6

The answer is:605.149963

***5. Write a recursive function that reads N and prints from N to 0.***

**PROGRAM:**

#include<stdio.h>

int range(int);

void main()

{

int n;

printf("enter the limit of the range you want");

scanf("%d",&n);

printf("the range is ");

range(n-1);

printf("\n");

}

int range(int i)

{

if(i>=0)

{

printf("%d",i);

return (0+range(i-1));

}

else

{

return 0;

}

}

**Output:**

cseb121@jtl-19:~$gcc revrange.c –o a2\_5

cseb121@jtl-19:~$./ a2\_5

enter the limit of the range you want:10

the range is 9876543210

***6. Factorials***

***The factorial of an integer n, written n!, is the product of all the integers from 1 to n inclusive. The factorial quickly becomes very large; 13! is too large to store as an integer on most computers, and 35! is too large for a floating-point variable. Your task is to find the rightmost non-zero digit of n!. (1<= n <= 100) For example, 5! = 1 \* 2 \* 3 \* 4 \* 5 = 120, so the rightmost non-zero digit of 5! is 2. Also, 7! = 1 \* 2 \* 3 \* 4 \* 5 \* 6 \* 7 = 5040, so the rightmost non-zero digit of 7! is 4.***

**PROGRAM:**

#include<stdio.h>

void main()

{

int n,result;

long int ans;

long int fact(int n);

int rmnzt(long int ans);

printf("Enter the number:");

scanf("%d",&n);

ans=fact(n);

result=rmnzt(ans);

printf("The result:%d\n",result);

}

long int fact(int n)

{

if (n==0)

{

return 1;

}

else

{

return(n\*fact(n-1));

}

}

int rmnzt(long int ans)

{

if (ans%10==0)

{

ans=ans/10;

rmnzt(ans);

}

else

{

return(ans%10);

}}

**Output:**

cseb121@jtl-19:~$gcc factright0.c –o a2\_6

cseb121@jtl-19:~$./ a2\_6

Enter the number:10

The result:8