

|  |
| --- |
| C.P.N.M. LAB REPORT |
|  |
| ASSIGNMENT 3  BCSE FIRST YEAR FIRST SEMESTER  Authored by: SOHAM CHOWDHURY |



**CPNM LAB ASSIGNMENT REPORT**

BCSE FIRST YEAR FIRST SEMESTER 2021-2022

NAME-SOHAM CHOWDHURY

DEPARTMENT-COMPUTER SCIENCE AND ENGINEERING

SECTION-A3.

ROLL NO-002110501145.

# ASSIGNMENT 3

1. Write a C program which accepts a number n and prints a. all integers divisible by n between 1 and 100 where value of n is provided by the user. b. all prime numbers between 1 and n. c. all prime factors of n. d. octal equivalent of n e. sum of digits. f. factorial of n. g. reverse of n.

HERE, a.i take a loop from one to hundred and take n from the user and if i%n==0 then print the number.

b.take n as input from user and then take a loop from 1 to n with variable I and check is I is prime or not,prime number is divisible only by one and itself.

c.take a number as input find its factors with the help of loop, check whether the factors are prime or not,if prime print them.

d.converted the decimal number inputted by user to octal number.

e.we used the logic of using digit extraction using modulus operator then storing the digits sum in a variable.

f.using loop from 1 to n and storing their product gives the factorial of a number.

g.reversed the number by digit extraction then multiplying them with 10.

#include<stdio.h>

int main()

{

    int n;

    char c;

    int di,num1=0;

    int d,sum=0;

    int rem,oct=0,num=0,digit;

    double product=1;

    printf("enter n=");

    scanf("%d",&n);

    fflush(stdin);

    printf("enter the choice = ");

    scanf("%c",&c);

    switch (c)

    {

        case 'a':

            for(int i=1;i<=100;i++)

            {

                if(i%n==0)

                printf("%d ",i);

            }

        break;

        case 'b':

            printf("the prime factors between 1 and %d are as follows ",n);

            for(int i=1;i<=n;i++)

            {

                int k=0;

                for(int j=2;j<i;j++)

                {

                    if(i%j==0)

                    k++;

                }

                if(k==0)

                printf("%d ",i);

            }

        break;

        case 'c':

            printf("the prime factors of %d are as as follows \n",n);

            for(int i=1;i<=n;i++)

            {

                if(n%i==0)

                {

                    int c=0;

                    for(int j=2;j<i;j++)

                    {

                        if(i%j==0)

                        c++;

                    }

                    if(c==0)

                    printf("%d\t",i);

                }

            }

        break;

        case 'd':

            printf("octal equivalent of %d is ",n);

                while(n!=0)

            {

                rem=n%8;

                n=n/8;

                num=num\*10+rem;

            }

            for(int i=num;i>0;i/=10)

            {

                digit=i%10;

                oct=oct\*10+digit;

            }

            printf("%d",oct);

        break;

        case 'e':

            for(int i=n;i>0;i/=10)

            {

                d=i%10;

                sum =sum+d;

            }

            printf("the sum of digits is %d",sum);

        break;

        case 'f':

            for(int i=1;i<=n;i++)

            {

            product=product\*i;

            }

            printf("the factorial of, %d! is %f",n,product);

        break;

        case 'g':

            for(int i=n;i>0;i/=10)

            {

                di=i%10;

                num1=num1\*10+di;

            }

            printf("the reverse of %d is %d",n,num1);

        break;

        default:

            printf("invalid choice");

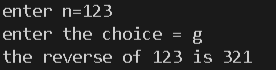
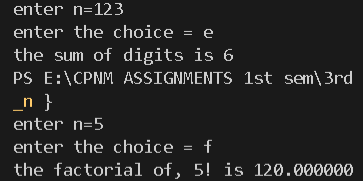
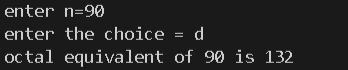
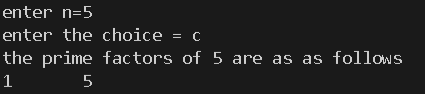
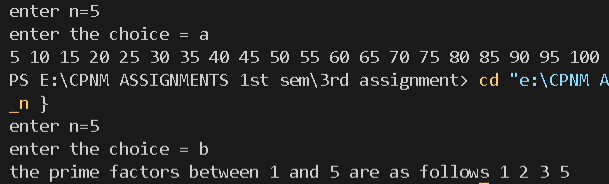
        break;

    }

    return 0;

}

OUTPUT-



1. Write a C program to find out the sum of the following series. a. S=1+2+3+4+ ... +n b. S=1.2+2.3+3.4+4.5+ ... +n.(n+1) c. S=1!+2!+3!+4!+ ... +n! d. S = 1@ + 2@ + 3@ + 4@ + … + n@ where, n@ is the sum of all factors of n. Example: 6@ = 1+2+3+6 = 12

My main objective here was to print all the series using loop and expressing each general term of each series in terms of the looping variable then storing their sum with each iteration of the loop.

Program:

#include<stdio.h>

int factorial(int a)

{

    if(a==0||a==1)

    return 1;

    else

    return a\*factorial(a-1);

}

int sumoffact(int a)

{

    int sum1=0;

    for(int i=1;i<=a;i++)

    {

        if(a%i==0)

        sum1=sum1+i;

    }

    return sum1;

}

int main()

{

    char c;

    printf("enter choice=");

    scanf("%c",&c);

    int n,sum=0;

    fflush(stdin);

    printf("enter n=");

    scanf("%d",&n);

    switch (c)

    {

        case 'a':

        for(int i=1;i<=n;i++)

        {

            sum=sum+i;

        }

        printf("the sum is %d",sum );

        break;

        case 'b':

        for(int i=1;i<=n;i++)

        {

            sum=sum+i\*(i+1);

        }

        printf("the sum is %d",sum );

        break;

        case 'c':

        for(int i=1;i<=n;i++)

        {

            int k=factorial(i);

            sum=sum+k;

        }

        printf("the sum is %d",sum);

        break;

        case 'd':

        for(int i=1;i<=n;i++)

        {

            int b=sumoffact(i);

            sum=sum+b;

        }

        printf("the sum is %d",sum);

        break;

        default:

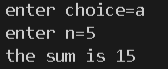
        printf("invalid choice ");

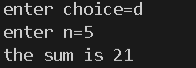
        break;

    }

    return 0;

}

Output: 



1. Write a program to generate all combinations of digit 1, 2 and 3 using a for loop.

I used the logic of three nested loops to generate all combinations of 1,2,3 such that none of them are same at the same time.

#include<stdio.h>

int main()

{

    printf("the combinations of digits 1,2,3 are as follows:-\n");

    for(int i=1;i<=3;i++)

    {

        for(int j=1;j<=3;j++)

        {

            for(int k=1;k<=3;k++)

            {

                if(i!=j&&j!=k&&i!=k)

                printf("%d%d%d\n",i,j,k);

            }

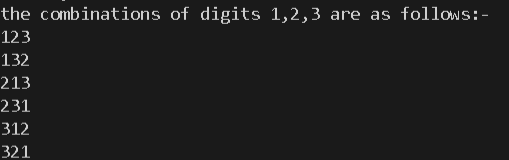
        }

    }

    return 0;

}

Output:



1. Write a menu driven program to accept a number in any Number System [from Binary, Decimal, Octal, and Hex] and convert and display the same in any other amongst these.

Here my main logic was to input the number from the user along with its base and the base of the resultant number.

First I converted the number to decimal then I converted it to the required base if required.

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int value(char c)

{

    if(c>='0'&&c<='9')

        return (int)c-'0';

    else

        return (int)c -'A'+10;

}

int any\_to\_dec(char \*str, int base)

{

    int len = strlen(str);

    int power = 1;

    int num = 0;

    int i;

    for(i=len-1;i>=0;i--)

    {

        num += value(str[i]) \* power;

        power = power \* base;

    }

    return num;

}

char reVal(int num)

{

    if(num >=0 && num <=9)

        return (char)(num + '0');

    else

        return (char)(num - 10 + 'A');

}

char\* dec\_to\_any( int inputNum, int base)

{

    static char res[100];

    int index=0;

    while(inputNum > 0)

    {

        res[index++] = reVal(inputNum % base);

        inputNum /= base;

    }

    res[index] = '\0';

    int len = strlen(res);

    int i;

    for(i=0;i<len/2;i++)

    {

        char temp = res[i];

        res[i] = res[len-i-1];

        res[len-i-1] =temp;

    }

    return res;

}

int main()

{

    int ibase,fbase;

    char \* ptr;

    printf("enter the initial base=");

    scanf("%d",&ibase);

    printf("enter the final base=");

    scanf("%d",&fbase);

    char str[100];

    fflush(stdin);

    printf("enter the number=");

    gets(str);

    char\* str1;

    int x,y,z;

    if(ibase==fbase)

    puts(str);

    else

    {

        int ch;

        if(ibase == 10)

        ch=1;

        else if(fbase == 10)

        ch=2;

        else

        ch=3;

        switch(ch)

        {

            case 1:

            x=atoi(str);

            str1=dec\_to\_any(x,fbase);

            printf("the resultant number is=");

            puts(str1);

            break;

            case 2:

            y=any\_to\_dec(str,ibase);

            printf("the resultant number is=%d",y);

            break;

            case 3:

            z=any\_to\_dec(str,ibase);

            ptr=dec\_to\_any(z,fbase);

            printf("the resultant number is=");

            puts(ptr);

            break;

            default:

            printf("invalid input");

            break;

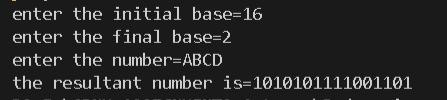
        }

    }

    return 0;

}

Output:



1. Write a program named SINE to find the sine of an angle. The angle and its unit (degree, radian or grade) should be provided as command line arguments. For the units, short forms as d/D (for degree), r/R (for radian) or g/G (for grade) may be used. The program should use the series sin 𝑠𝑖𝑛 (𝑥) = 𝑥 − , 𝑥 3 3! + 𝑥 5 5! − ⋯ for evaluation. Take care of negative angles and angles in all the quadrants.

Here I took the input through command line arguments as per mentioned in the question 2nd and 3rd arguments were value and unit respectively

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

#include<string.h>

#define PI 3.14159265

double factorial(int n)

{

    if(n==0||n==1)

    return 1;

    else

    return n\*factorial(n-1);

}

int main(int argc,char const\* argv[])

{

    if(argc==3)

    {

        int x;

        double sum=0;

        float rad;

        x = atoi(argv[1]);

        printf("the given angle value is=%d\n",x);

        if(\*argv[2]=='r'||\*argv[2]=='R')

        rad=x;

        else if(\*argv[2]=='d'||\*argv[2]=='D')

        rad=PI\*x/180;

        else if(\*argv[2]=='g'||\*argv[2]=='G')

        rad=x\*0.015708;

        for(int i=1;i<=100;i++)

        {

            sum=sum+(double)(pow(-1,i-1)\*pow(rad,2\*i-1)/factorial(2\*i-1));

        }

        printf("sin(%d%s) = %0.3f",x,argv[2],sum);

    }

    else

    {

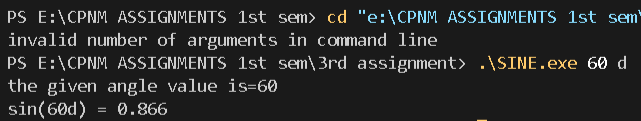
        printf("invalid number of arguments in command line");

    }

    return 0;

}

Output:



1. Write a C program to print the first n numbers of the Fibonacci sequence. The Fibonacci sequence is constructed by adding the last two numbers of the sequence so far to get the next number in the sequence. The first and second numbers of the sequence are defined as 0 and 1. We get: 0, 1, 1, 2, 3, 5, 8, 13, 21…

Using loop and assuming first element to be 0 and 1 , I printed the Fibonacci series in which the n th term is the sum of n-1th term and n-2 th term.

#include<stdio.h>

int main()

{

    int n,sum;

    printf("enter n=");

    scanf("%d",&n);

    int a=0,b=1;

    if(n>=2)

    {

        printf("%d,%d",a,b);

        for(int i=1;i<n-1;i++)

        {

            sum=a+b;

            printf(",%d",sum);

            a=b;

            b=sum;

        }

    }

    else if(n==1)

    printf("%d",a);

    else

    printf("invalid number of terms");

    return 0;

}

Output:



1. Write a program to print out all Armstrong numbers between 1 and 500. If the sum of cubes of each digit of the number is equal to the number itself, then the number is called an Armstrong number. For example, 153 = ( 1 3 ) + ( 5 3 ) + ( 3 3 ).

Using loop I extracted each and every digit of the number given by the user then stored their cube in the sum variable and if it is equivalent to the number itself then it is an Armstrong number.

#include<stdio.h>

#include<math.h>

int main()

{

    int d;

    printf("the armstrong numbers between 1 and 500 are as follows :-\n");

    for(int i=1;i<=500;i++)

    {

        int sum=0;

        for(int j=i;j>0;j=j/10)

        {

            d=j%10;

            sum=sum+d\*d\*d;

        }

        if(sum==i)

        printf("%d\t",i);

    }

    return 0;

}

Output:



1. Write a C program which prints the first 10 happy numbers. If you iterate the process (assume maximum 100 iterations) of summing the squares of the decimal digits of a number and if the process terminates in 1, then the original number is called a Happy number. For example 7 is a happy number as 7 → 49 → 97 → 130 → 10 → 1.

Here I used the principles of digit extraction then storing sum of the squares of the digit in another variable and made that the new number until it becomes single digit and if itn is one then the number is happy.

#include<stdio.h>

#include<stdio.h>

#include<math.h>

void main()

{

    int n=1,num=10,d;

    printf("the 1st 10 happy numbers are as follows:-\n");

    for(int c=0;c<10;)

    {

        num=n;

        do

        {

            int sum=0;

            for(int i=num;i>0;i=i/10)

            {

                d=i%10;

                sum=sum+d\*d;

            }

            num=sum;

        }while (num>9);

        if(num==1)

        {

            printf("%d\t",n);

            c++;

        }

        n++;

    }

}

Output:



1. An important property of square numbers: If a natural number is a square number, then it has to be the sum of Successive Odd Numbers starting from 1. For example: Perfect Square Sum of Odd

Sum of first n odd numbers is equal to the square of the number n.

#include<stdio.h>

#include<math.h>

int main()

{

    int n,c=0,sum=0;

    printf("enter the perfect square number = ");

    scanf("%d",&n);

    for(int i=1;i<n;i=i+2)

    {

        sum=sum+i;

        c++;

        if(sum==n)

        break;

    }

    printf("the square root of %d is = %d",n,c);//sum of n consecutive numbers = n\*n

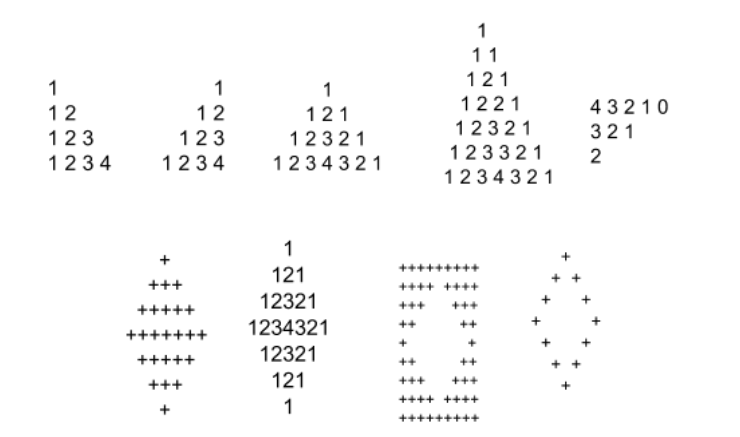
    return 0;

}

Ouput:



1. W rite a C program that prints the following patterns for the input of n = 4. T he value of n is input by the user



In all the 9 types of patterns I devided the pattern into multiple parts and I used the looping variable i from row tracking j for column tracking also used spaces and starts where it was necessary. PROGRAMS ARE IN ORDER OF TOP TO BOTTOM AND LEFT TO RIGHT SIMULTANEOUSLY.

Program:

#include<stdio.h>

#include<math.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    for(int i=1;i<=n;i++)

    {

        for(int j=1;j<=i;j++)

        {

            printf("%d",j);

        }

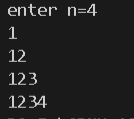
        printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

#include<math.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    for(int i=1;i<=n;i++)

    {

        for(int j=1;j<=n-i;j++)

        {

            printf(" ");

        }

        for(int j=1;j<=i;j++)

        {

            printf("%d",j);

        }

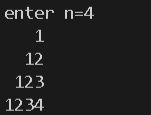
        printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

#include<math.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    for(int i=1;i<=n;i++)

    {

        for(int j=0;j<=n-i;j++)

        {

            printf(" ");

        }

        for(int j=1;j<=i;j++)

        {

            printf("%d",j);

        }

        if(i>=2)

        {

            for(int k=i-1;k>=1;k--)

            {

                printf("%d",k);

            }

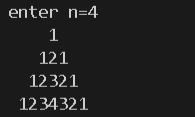
        }

        printf("\n");

    }

    return 0;

}

Output: 

Program:

#include<stdio.h>

#include<math.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    int k=n-1;

    for(int i=1;i<=2\*n-1;i++)

    {

        if(i<=n)

        {

            for(int j=1;j<=n-i;j++)

            printf(" ");

            for(int j=1;j<=2\*i-1;j=j+1)

            printf("\*");

        }

        else

        {

            for(int j=1;j<=i-n;j++)

            printf(" ");

            for(int j=1;j<=2\*k-1;j++)

            printf("\*");

            if(k==1)

            break;

            k--;

        }

        printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    int b=0;

    for(int i=1;i<=2\*n-1;i++)

    {

        if(i<=n)

        {

            for(int j=1;j<=n-i;j++)

            printf(" ");

            for(int j=1;j<=i;j++)

            printf("%d",j);

            if(i>=2)

            {

                for(int k=i-1;k>=1;k--)

                printf("%d",k);

            }

        }

        else if(i>n)

        {

            for(int j=1;j<=i-n;j++)

            printf(" ");

            for(int k=1;k<=2\*n-i;k++)

            printf("%d",k);

            for(int c=2\*n-i-1;c>=1;c--)

            printf("%d",c);

        }

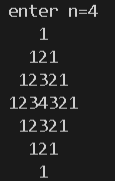
        printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

int main()

{

    int n,d=2;

    printf("enter n=");

    scanf("%d",&n);

    int c=n,f=2\*n-3;

    for(int i=1;i<=2\*n+1;i++)

    {

            if(i==1||i==2\*n+1)

            {

                for(int j=1;j<=2\*n+1;j++)

                printf("\*");

            }

            if(i>=2&&i<=n+1)

            {

                for(int j=1;j<=c;j++)

                printf("\*");

                for(int k=1;k<=2\*i-3;k++)

                printf(" ");

                for(int j=1;j<=c;j++)

                printf("\*");

                c--;

            }

            if(i>=n+2&&i<=2\*n)

            {

                for(int j=1;j<=d;j++)

                printf("\*");

                for(int j=1;j<=f;j++)

                printf(" ");

                f=f-2;

                for(int j=1;j<=d;j++)

                printf("\*");

                d++;

            }

            printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

int main()

{

    int n;

    printf("enter n=");

    scanf("%d",&n);

    int d=2\*n-5;

    for(int i=1;i<=2\*n-1;i++)

    {

        if(i<=n)

        {

            for(int j=1;j<=n-i;j++)

            {

                printf(" ");

            }

            printf("\*");

            for(int j=1;j<=2\*i-3;j++)

            printf(" ");

            if(i>=2&&i<=n)

            {

                printf("\*");

            }

        }

        else

        {

            for(int j=1;j<=i-n;j++)

            printf(" ");

            printf("\*");

            for(int j=1;j<=d;j++)

            printf(" ");

            d=d-2;

            if(i<2\*n-1)

            {

                printf("\*");

            }

        }

        printf("\n");

    }

    return 0;

}

Output:



Program:

#include<stdio.h>

int main()

{

    int n,k;

    printf("enter n=");

    scanf("%d",&n);

    for(int i=1;i<=2\*n-1;i++)

    {

        if(i%2!=0)

        {

            int k=(i+1)/2;

            for(int j=1;j<=n-k;j++)

            printf("  ");

            for(int j=1;j<=k;j++)

            printf("%d ",j);

            for(int j=k-1;j>=1;j--)

            printf("%d ",j);

        }

        else

        {

            int d=i/2;

            for(int j=2\*n-1-i;j>=1;j--)

            printf(" ");

            for(int j=1;j<=d;j++)

            printf("%d ",j);

            for(int j=d;j>=1;j--)

            printf("%d ",j);

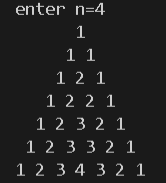
        }

        printf("\n");

    }

    return 0;

}

Output: 

Program:

#include<stdio.h>

int main()

{

    int n,c,d;

    printf("enter n=");

    scanf("%d",&n);

    c=n;

    d=c;

    for(int i=1;i<=n-1;i++)

    {

        for(int j=1;j<=2\*n-1-2\*i;j++)

        {

            printf("%d",c);

            c--;

        }

        d--;

        c=d;

        printf("\n");

    }

    return 0;

}

Output:

