**INTRODUCTION TO PROGRAMMING IN C**



GITHUB ASSIGNMENT

**SUBMITTED BY:**

Ojaswin Kohli

CSE7-INTERNATIONAL

A023109520005

\*\*

/\* C program to all natural numbers in reverse from n to 1

\*/

#include <stdio.h>

int main()

{

int i, start;

/\* Input start range from user \*/

printf("Enter starting value: ");

scanf("%d", &start);

/\*

\* Run loop from 'start' to 1 and

\* decrement 1 in each iteration

\*/

for(i=start; i>=1; i--)

{

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

}

return 0;

}

/\*\*

\* C program to calculate factorial of a number

\*/

#include <stdio.h>

int main()

{

int i, num;

unsigned long long fact=1LL;

/\* Input number from user \*/

printf("Enter any number to calculate factorial: ");

scanf("%d", &num);

/\* Run loop from 1 to num \*/

for(i=1; i<=num; i++)

{

fact = fact \* i;

}

printf("Factorial of %d = %llu", num, fact);

return 0;

}

/\*\*

\* C program to calculate gross salary of an employee

\*/

#include <stdio.h>

int main()

{

float basic, gross, da, hra;

/\* Input basic salary of employee \*/

printf("Enter basic salary of an employee: ");

scanf("%f", &basic);

/\* Calculate D.A and H.R.A according to specified conditions \*/

if(basic <= 10000)

{

da = basic \* 0.8;

hra = basic \* 0.2;

}

else if(basic <= 20000)

{

da = basic \* 0.9;

hra = basic \* 0.25;

}

else

{

da = basic \* 0.95;

hra = basic \* 0.3;

}

/\* Calculate gross salary \*/

gross = basic + hra + da;

printf("GROSS SALARY OF EMPLOYEE = %.2f", gross);

return 0;

}

/\*\*

\* C program to calculate product of digits of a number

\*/

#include <stdio.h>

int main()

{

int num;

long long product=1ll;

/\* Input number from user \*/

printf("Enter any number to calculate product of digit: ");

scanf("%d", &num);

product = (num == 0 ? 0 : 1ll);

/\* Repeat the steps till num becomes 0 \*/

while(num != 0)

{

/\* Get the last digit from num and multiplies to product \*/

product = product \* (num % 10);

/\* Remove the last digit from n \*/

num = num / 10;

}

printf("Product of digits = %lld", product);

return 0;

}

/\*\*

\* C program to calculate profit or loss

\*/

#include <stdio.h>

int main()

{

int cp,sp, amt;

/\* Input cost price and selling price of a product \*/

printf("Enter cost price: ");

scanf("%d", &cp);

printf("Enter selling price: ");

scanf("%d", &sp);

if(sp > cp)

{

/\* Calculate Profit \*/

amt = sp - cp;

printf("Profit = %d", amt);

}

else if(cp > sp)

{

/\* Calculate Loss \*/

amt = cp - sp;

printf("Loss = %d", amt);

}

else

{

/\*\*

\* C program to calculate sum of digits using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

int sumOfDigits(int num);

int main()

{

int num, sum;

printf("Enter any number to find sum of digits: ");

scanf("%d", &num);

sum = sumOfDigits(num);

printf("Sum of digits of %d = %d", num, sum);

return 0;

}

/\*\*

\* Recursive function to find sum of digits of a number

\*/

int sumOfDigits(int num)

{

// Base condition

if(num == 0)

return 0;

return ((num % 10) + sumOfDigits(num / 10));

}

/\*\*

\* C program to calculate total electricity bill

\*/

#include <stdio.h>

int main()

{

int unit;

float amt, total\_amt, sur\_charge;

/\* Input unit consumed from user \*/

printf("Enter total units consumed: ");

scanf("%d", &unit);

/\* Calculate electricity bill according to given conditions \*/

if(unit <= 50)

{

amt = unit \* 0.50;

}

else if(unit <= 150)

{

amt = 25 + ((unit-50) \* 0.75);

}

else if(unit <= 250)

{

amt = 100 + ((unit-150) \* 1.20);

}

else

{

amt = 220 + ((unit-250) \* 1.50);

}

/\*

\* Calculate total electricity bill

\* after adding surcharge

\*/

sur\_charge = amt \* 0.20;

total\_amt = amt + sur\_charge;

printf("Electricity Bill = Rs. %.2f", total\_amt);

return 0;

}

/\*\*

\* C program to check alphabet, digit or special character

\*/

#include <stdio.h>

int main()

{

char ch;

/\* Input character from user \*/

printf("Enter any character: ");

scanf("%c", &ch);

/\* Alphabet check \*/

if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))

{

printf("'%c' is alphabet.", ch);

}

else if(ch >= '0' && ch <= '9')

{

printf("'%c' is digit.", ch);

}

else

{

printf("'%c' is special character.", ch);

}

return 0;

}

#include<stdio.h>

int main()

{

int n,r,sum=0,temp;

printf("enter the number=");

scanf("%d",&n);

temp=n;

while(n>0)

{

r=n%10;

sum=sum+(r\*r\*r);

n=n/10;

}

if(temp==sum)

printf("armstrong number ");

else

printf("not armstrong number");

return 0;

}

/\*\*

\* C program to check divisibility of any number

\*/

#include <stdio.h>

int main()

{

int num;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &num);

/\*

\* If num modulo division 5 is 0

\* and num modulo division 11 is 0 then

\* the number is divisible by 5 and 11 both

\*/

if((num % 5 == 0) && (num % 11 == 0))

{

printf("Number is divisible by 5 and 11");

}

else

{

printf("Number is not divisible by 5 and 11");

}

return 0;

/\*\*

\* C program to check even or odd number

\*/

#include <stdio.h>

int main()

{

int num;

/\* Input number from user \*/

printf("Enter any number to check even or odd: ");

scanf("%d", &num);

/\* Check if the number is divisible by 2 then it is even \*/

if(num % 2 == 0)

{

/\* num % 2 is 0 \*/

printf("Number is Even.");

}

else

{

/\* num % 2 is 1 \*/

printf("Number is Odd.");

}

return 0;

}

/\*\*

\* C program to check even or odd using functions

\*/

#include <stdio.h>

/\*\*

\* Function to check even or odd

\* Returns 1 is num is even otherwise 0

\*/

int isEven(int num)

{

return !(num & 1);

}

int main()

{

int num;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &num);

/\* If isEven() function returns 0 then the number is even \*/

if(isEven(num))

{

printf("The number is even.");

}

else

{

printf("The number is odd.");

}

return 0;

}

/\*\*

\* C program to check Leap Year

\*/

#include <stdio.h>

int main()

{

int year;

/\* Input year from user \*/

printf("Enter year : ");

scanf("%d", &year);

/\*

\* If year is exactly divisible by 4 and year is not divisible by 100

\* or year is exactly divisible by 400 then

\* the year is leap year.

\* Else year is normal year

\*/

if(((year % 4 == 0) && (year % 100 !=0)) || (year % 400==0))

{

printf("LEAP YEAR");

}

else

{

printf("COMMON YEAR");

}

return 0;

}

/\*\*

\* C program to check palindrome number using recursion

\*/

#include <stdio.h>

#include <math.h>

/\* Function declarations \*/

int reverse(int num);

int isPalindrome(int num);

/\*\*

\* C program to check positive negative or zero using if else

\*/

#include <stdio.h>

int main()

{

int num;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &num);

if(num > 0)

{

printf("Number is POSITIVE");

}

else if(num < 0)

{

printf("Number is NEGATIVE");

}

else

{

printf("Number is ZERO");

}

return 0;

}

/\*\*

\* C program to check whether a character is alphabet or not

\*/

#include <stdio.h>

int main()

{

char ch;

/\* Input a character from user \*/

printf("Enter any character: ");

scanf("%c", &ch);

if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))

{

printf("Character is an ALPHABET.");

}

else

{

printf("Character is NOT ALPHABET.");

}

return 0;

/\*\*

\* C program to check whether a character is uppercase or lowercase

\*/

#include <stdio.h>

int main()

{

char ch;

/\* Input character from user \*/

printf("Enter any character: ");

scanf("%c", &ch);

if(ch >= 'A' && ch <= 'Z')

{

printf("'%c' is uppercase alphabet.", ch);

}

else if(ch >= 'a' && ch <= 'z')

{

printf("'%c' is lowercase alphabet.", ch);

}

else

{

printf("'%c' is not an alphabet.", ch);

}

return 0;

}

/\*\*

\* C program to check whether a character is vowel or consonant

\*/

#include <stdio.h>

int main()

{

char ch;

/\* Input character from user \*/

printf("Enter any character: ");

scanf("%c", &ch);

/\* Condition for vowel \*/

if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u' ||

ch=='A' || ch=='E' || ch=='I' || ch=='O' || ch=='U')

{

printf("'%c' is Vowel.", ch);

}

else if((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))

{

/\* Condition for consonant \*/

printf("'%c' is Consonant.", ch);

}

else

{

/\*

\* If it is neither vowel nor consonant

\* then it is not an alphabet.

\*/

printf("'%c' is not an alphabet.", ch);

}

return 0;

}

/\*\*

\* C program to check whether a number is palindrome or not

\*/

#include <stdio.h>

int main()

{

int n, num, rev = 0;

/\* Input a number from user \*/

printf("Enter any number to check palindrome: ");

scanf("%d", &n);

/\* Copy original value to 'num' to 'n'\*/

num = n;

/\* Find reverse of n and store in rev \*/

while(n != 0)

{

rev = (rev \* 10) + (n % 10);

n /= 10;

}

/\* Check if reverse is equal to 'num' or not \*/

if(rev == num)

{

printf("%d is palindrome.", num);

}

else

{

printf("%d is not palindrome.", num);

}

return 0;

}

/\*\*

\* C program to check whether a number is Perfect number or not

\*/

#include <stdio.h>

int main()

{

int i, num, sum = 0;

/\* Input a number from user \*/

printf("Enter any number to check perfect number: ");

scanf("%d", &num);

/\* Calculate sum of all proper divisors \*/

for(i = 1; i <= num / 2; i++)

{

/\* If i is a divisor of num \*/

if(num%i == 0)

{

sum += i;

}

}

/\* Check whether the sum of proper divisors is equal to num \*/

if(sum == num)

{

printf("%d is PERFECT NUMBER", num);

}

else

{

printf("%d is NOT PERFECT NUMBER", num);

}

return 0;

}

/\*\*

\* C program to check whether a number is Strong Number or not

\*/

#include <stdio.h>

int main()

{

int i, originalNum, num, lastDigit, sum;

long fact;

/\* Input a number from user \*/

printf("Enter any number to check Strong number: ");

scanf("%d", &num);

/\* Copy the value of num to a temporary variable \*/

originalNum = num;

sum = 0;

/\* Find sum of factorial of digits \*/

while(num > 0)

{

/\* Get last digit of num \*/

lastDigit = num % 10;

/\* Find factorial of last digit \*/

fact = 1;

for(i=1; i<=lastDigit; i++)

{

fact = fact \* i;

}

/\* Add factorial to sum \*/

sum = sum + fact;

num = num / 10;

}

/\* Check Strong number condition \*/

if(sum == originalNum)

{

printf("%d is STRONG NUMBER", originalNum);

}

else

{

printf("%d is NOT STRONG NUMBER", originalNum);

}

return 0;

}

/\*\*

\* C program to check whether a triangle is Equilateral, Isosceles or Scalene

\*/

#include <stdio.h>

int main()

{

int side1, side2, side3;

/\* Input sides of a triangle \*/

printf("Enter three sides of triangle: ");

scanf("%d%d%d", &side1, &side2, &side3);

if(side1==side2 && side2==side3)

{

/\* If all sides are equal \*/

printf("Equilateral triangle.");

}

else if(side1==side2 || side1==side3 || side2==side3)

{

/\* If any two sides are equal \*/

printf("Isosceles triangle.");

}

else

{

/\* If none sides are equal \*/

printf("Scalene triangle.");

}

return 0;

}

/\*\*

\* C program to check whether a triangle is valid or not if angles are given

\*/

#include <stdio.h>

int main()

{

int angle1, angle2, angle3, sum;

/\* Input all three angles of triangle \*/

printf("Enter three angles of triangle: \n");

scanf("%d%d%d", &angle1, &angle2, &angle3);

/\* Calculate sum of angles \*/

sum = angle1 + angle2 + angle3;

/\*

\* If sum of angles is 180 and

\* angle1, angle2, angle3 is not 0 then

\* triangle is valid.

\*/

if(sum == 180 && angle1 > 0 && angle2 > 0 && angle3 > 0)

{

printf("Triangle is valid.");

}

else

{

printf("Triangle is not valid.");

}

return 0;

}

/\*\*

\* C program to check whether a triangle is valid or not if its sides are given

\*/

#include <stdio.h>

int main()

{

int side1, side2, side3;

/\* Input three sides of a triangle \*/

printf("Enter three sides of triangle: \n");

scanf("%d%d%d", &side1, &side2, &side3);

if((side1 + side2) > side3)

{

if((side2 + side3) > side1)

{

if((side1 + side3) > side2)

{

/\*

\* If side1 + side2 > side3 and

\* side2 + side3 > side1 and

\* side1 + side3 > side2 then

\* the triangle is valid.

\*/

printf("Triangle is valid.");

}

else

{

printf("Triangle is not valid.");

}

}

else

{

printf("Triangle is not valid.");

}

}

else

{

printf("Triangle is not valid.");

}

return 0;

}

/\*\*

\* C program to convert binary to decimal number system

\*/

#include <stdio.h>

#include <math.h>

#include <string.h>

#define SIZE 16 /\* N-bit binary number \*/

void getTwosComplement(char \*twosComp, const char \*);

int main()

{

char binary[SIZE + 1], tempBinary[SIZE + 1];

int i, j, signBit = 0;

long long decimal = 0;

printf("Enter any %d bit binary value: ", SIZE);

gets(binary);

strcpy(tempBinary, binary);

/\*

\* If sign bit is on find two's complement of the binary number

\*/

if(binary[0] == '1')

{

signBit = 1;

getTwosComplement(tempBinary, binary);

}

/\*

\* Convert decimal to binary number

\*/

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

{

if(tempBinary[i] == '1')

{

decimal += pow(2, (SIZE - (i+1)));

}

}

if(signBit==1)

{

decimal \*= -1;

}

printf("Binary value = %s\n", binary);

printf("Decimal value = %lld", decimal);

return 0;

}

/\*

\* Gets the 2's complement of the binary value.

\*/

void getTwosComplement(char \* twosComp, const char \* binary)

{

char onesComp[SIZE + 1];

int i, carry=1;

/\*

\* Finds 1's complement of the binary number

\*/

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

{

if(binary[i]=='1')

{

onesComp[i] = '0';

}

else if(binary[i]=='0')

{

onesComp[i] = '1';

}

}

onesComp[SIZE] = '\0';

/\*

\* Adds 1 to 1's complement of the binary number to get 2's complement

\*/

for(i=SIZE-1; i>=0; i--)

{

if(onesComp[i]=='1' && carry==1)

{

twosComp[i] = '0';

}

else if(onesComp[i]=='0' && carry==1)

{

twosComp[i] = '1';

carry = 0;

}

else

{

twosComp[i] = onesComp[i];

}

}

twosComp[SIZE] = '\0';

}

/\*\*

\* C program to convert binary to hexadecimal number system

\*/

#include <stdio.h>

#include <string.h>

int main()

{

int hexConstant[] = {0, 1, 10, 11, 100, 101, 110, 111, 1000,

1001, 1010, 1011, 1100, 1101, 1110, 1111};

long long binary, tempBinary;

char hex[20];

int index, i, digit;

/\* Input binary number from user \*/

printf("Enter binary number: ");

scanf("%lld", &binary);

/\* Copy binary number to temp variable \*/

tempBinary = binary;

index = 0;

/\* Find hexadecimal of binary number \*/

while(tempBinary!=0)

{

/\* Group binary to last four digits \*/

digit = tempBinary % 10000;

/\* Find hexadecimal equivalent of last four digit \*/

for(i=0; i<16; i++)

{

if(hexConstant[i] == digit)

{

if(i<10)

{

/\* 0-9 integer constant \*/

hex[index] = (char)(i + 48);

}

else

{

/\* A-F character constant \*/

hex[index] = (char)((i-10) + 65);

}

index++;

break;

}

}

/\* Remove the last 4 digits as it is processed \*/

tempBinary /= 10000;

}

hex[index] = '\0';

/\* Reverse the hex digits \*/

strrev(hex);

printf("Binary number = %lld\n", binary);

printf("Hexadecimal number = %s", hex);

return 0;

}

/\*\*

\* C program to convert binary to octal number system

\*/

#include <stdio.h>

int main()

{

int octalConstant[] = {0, 1, 10, 11, 100, 101, 110, 111};

long long binary, octal, tempBinary;

int digit, place, i;

octal = 0;

place= 1;

/\* Input binary number from user \*/

printf("Enter any binary number: ");

scanf("%lld", &binary);

/\* Copy original binary value to temp variable \*/

tempBinary = binary;

while(tempBinary != 0)

{

/\* Extract last three digit of binary \*/

digit = tempBinary % 1000;

/\* Find octal equivalent of 3 digit binary \*/

for(i=0; i<8; i++)

{

if(octalConstant[i] == digit)

{

/\*

\* Increase the place value of octal

\* and add the previous octal value

\*/

octal = (i \* place) + octal;

break;

}

}

/\* Remove the last three digit of binary \*/

tempBinary /= 1000;

/\* Increase the place value \*/

place \*= 10;

}

printf("Original binary number = %lld\n", binary);

printf("Octal number = %lld", octal);

return 0;

}

/\*\*

\* C program to convert from Decimal number system to hexadecimal number system

\*/

#include <stdio.h>

#include <string.h>

int main()

{

char HEXVALUE[] = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F'};

long long decimal, tempDecimal;

char hex[65];

int index, rem;

/\* Input decimal number from user \*/

printf("Enter any decimal number: ");

scanf("%lld", &decimal);

tempDecimal = decimal;

index = 0;

/\* Decimal to hexadecimal conversion \*/

while(tempDecimal !=0)

{

rem = tempDecimal % 16;

hex[index] = HEXVALUE[rem];

tempDecimal /= 16;

index++;

}

hex[index] = '\0';

strrev(hex);

printf("\nDecimal number = %lld\n", decimal);

printf("Hexadecimal number = %s", hex);

return 0;

}

/\*\*

\* C program to convert from Decimal to Binary number system

\*/

#include <stdio.h>

#include <string.h>

int main()

{

long decimal, tempDecimal;

char binary[65];

int index = 0;

/\* Input decimal number from user \*/

printf("Enter any decimal value : ");

scanf("%ld", &decimal);

/\* Copy decimal value to temp variable \*/

tempDecimal = decimal;

while(tempDecimal > 0)

{

binary[index] = (tempDecimal % 2) + '0';

tempDecimal /= 2;

index++;

}

binary[index] = '\0';

/\* Reverse the converted binary to get final binary result \*/

strrev(binary);

printf("Decimal value = %ld\n", decimal);

printf("Binary value of decimal = %s", binary);

return 0;

}

/\*\*

\* C program to convert from Decimal to Octal number system

\*/

#include <stdio.h>

int main()

{

long long decimal, tempDecimal, octal;

int i, rem, place = 1;

octal = 0;

/\* Input decimal number from user \*/

printf("Enter any decimal number: ");

scanf("%lld", &decimal);

tempDecimal = decimal;

/\* Decimal to octal conversion \*/

while(tempDecimal > 0)

{

rem = tempDecimal % 8;

octal = (rem \* place) + octal;

tempDecimal /= 8;

place \*= 10;

}

printf("\nDecimal number = %lld\n", decimal);

printf("Octal number = %lld", octal);

return 0;

}

/\*\*

\* C program to convert Hexadecimal to binary number system

\*/

#include <stdio.h>

#include <string.h>

int main()

{

char hex[17], bin[65] = "";

int i = 0;

/\* Input hexadecimal number from user \*/

printf("Enter any hexadecimal number: ");

gets(hex);

/\* Extract first digit and find binary of each hex digit \*/

for(i=0; hex[i]!='\0'; i++)

{

switch(hex[i])

{

case '0':

strcat(bin, "0000");

break;

case '1':

strcat(bin, "0001");

break;

case '2':

strcat(bin, "0010");

break;

case '3':

strcat(bin, "0011");

break;

case '4':

strcat(bin, "0100");

break;

case '5':

strcat(bin, "0101");

break;

case '6':

strcat(bin, "0110");

break;

case '7':

strcat(bin, "0111");

break;

case '8':

strcat(bin, "1000");

break;

case '9':

strcat(bin, "1001");

break;

case 'a':

case 'A':

strcat(bin, "1010");

break;

case 'b':

case 'B':

strcat(bin, "1011");

break;

case 'c':

case 'C':

strcat(bin, "1100");

break;

case 'd':

case 'D':

strcat(bin, "1101");

break;

case 'e':

case 'E':

strcat(bin, "1110");

break;

case 'f':

case 'F':

strcat(bin, "1111");

break;

default:

printf("Invalid hexadecimal input.");

}

}

printf("Hexademial number = %s\n", hex);

printf("Binary number = %s", bin);

return 0;

}

/\*\*

\* C program to convert Hexadecimal to Decimal number system

\*/

#include <stdio.h>

#include <math.h>

#include <string.h>

int main()

{

char hex[17];

long long decimal, place;

int i = 0, val, len;

decimal = 0;

place = 1;

/\* Input hexadecimal number from user \*/

printf("Enter any hexadecimal number: ");

gets(hex);

/\* Find the length of total number of hex digit \*/

len = strlen(hex);

len--;

/\*

\* Iterate over each hex digit

\*/

for(i=0; hex[i]!='\0'; i++)

{

/\* Find the decimal representation of hex[i] \*/

if(hex[i]>='0' && hex[i]<='9')

{

val = hex[i] - 48;

}

else if(hex[i]>='a' && hex[i]<='f')

{

val = hex[i] - 97 + 10;

}

else if(hex[i]>='A' && hex[i]<='F')

{

val = hex[i] - 65 + 10;

}

decimal += val \* pow(16, len);

len--;

}

printf("Hexadecimal number = %s\n", hex);

printf("Decimal number = %lld", decimal);

return 0;

}

/\*\*

\* C program to convert Hexadecimal to Octal number system

\*/

#include <stdio.h>

int main()

{

char hex[17];

long long octal, bin, place;

int i = 0, rem, val;

/\* Input hexadecimal number from user \*/

printf("Enter any hexadecimal number: ");

gets(hex);

octal = 0ll;

bin = 0ll;

place = 0ll;

/\* Hexadecimal to binary conversion \*/

for(i=0; hex[i]!='\0'; i++)

{

bin = bin \* place;

switch(hex[i])

{

case '0':

bin += 0;

break;

case '1':

bin += 1;

break;

case '2':

bin += 10;

break;

case '3':

bin += 11;

break;

case '4':

bin += 100;

break;

case '5':

bin += 101;

break;

case '6':

bin += 110;

break;

case '7':

bin += 111;

break;

case '8':

bin += 1000;

break;

case '9':

bin += 1001;

break;

case 'a':

case 'A':

bin += 1010;

break;

case 'b':

case 'B':

bin += 1011;

break;

case 'c':

case 'C':

bin += 1100;

break;

case 'd':

case 'D':

bin += 1101;

break;

case 'e':

case 'E':

bin += 1110;

break;

case 'f':

case 'F':

bin += 1111;

break;

default:

printf("Invalid hexadecimal input.");

}

place = 10000;

}

place = 1;

/\* Binary to octal conversion \*/

while(bin > 0)

{

rem = bin % 1000;

switch(rem)

{

case 0:

val = 0;

break;

case 1:

val = 1;

break;

case 10:

val = 2;

break;

case 11:

val = 3;

break;

case 100:

val = 4;

break;

case 101:

val = 5;

break;

case 110:

val = 6;

break;

case 111:

val = 7;

break;

}

octal = (val \* place) + octal;

bin /= 1000;

place \*= 10;

}

printf("Hexadecimal number = %s\n", hex);

printf("Octal number = %lld", octal);

return 0;

}

/\*\*

\* C program to convert Octal number system to Binary number system

\*/

#include <stdio.h>

int main()

{

int OCTALVALUES[] = {0, 1, 10, 11, 100, 101, 110, 111};

long long octal, tempOctal, binary, place;

int rem;

/\* Input Octal number from user \*/

printf("Enter any Octal number: ");

scanf("%lld", &octal);

tempOctal = octal;

binary = 0;

place = 1;

/\* Convert octal to binary \*/

while(tempOctal > 0)

{

/\* Extract the last digit of octal \*/

rem = tempOctal % 10;

/\*

\* Get the binary equivalent of octal digit

\* add it to the binary variable

\*/

binary = (OCTALVALUES[rem] \* place) + binary;

/\* Remove the last octal digit \*/

tempOctal /= 10;

/\* Increase the place value \*/

place \*= 1000;

}

printf("Octal number = %lld\n", octal);

printf("Binary number = %lld", binary);

return 0;

}

/\*\*

\* C program to convert Octal number system to Decimal number system

\*/

#include <stdio.h>

#include <math.h>

int main()

{

long long octal, tempOctal, decimal;

int rem, place;

/\* Input octal number from user \*/

printf("Enter any octal number: ");

scanf("%lld", &octal);

tempOctal = octal;

decimal = 0;

place = 0;

/\*

\* Convert octal to decimal

\*/

while(tempOctal > 0)

{

/\* Extract the last digit of octal \*/

rem = tempOctal % 10;

/\* Convert last octal digit to decimal \*/

decimal += pow(8, place) \* rem;

/\* Remove the last octal digit \*/

tempOctal /= 10;

place++;

}

printf("Octal number = %lld\n", octal);

printf("Decimal number = %lld", decimal);

return 0;

}

/\*\*

\* C program to convert Octal to Hexadecimal number system

\*/

#include <stdio.h>

int main()

{

int OCTALVALUES[] = {0, 1, 10, 11, 100, 101, 110, 111};

long long octal, tempOctal, binary, place;

char hex[65] = "";

int rem;

place = 1;

binary = 0;

/\* Input octal number from user \*/

printf("Enter any octal number: ");

scanf("%lld", &octal);

tempOctal = octal;

/\*

\* Octal to binary conversion

\*/

while(tempOctal > 0)

{

rem = tempOctal % 10;

binary = (OCTALVALUES[rem] \* place) + binary;

tempOctal /= 10;

place \*= 1000;

}

/\*

\* Binary to hexadecimal conversion

\*/

while(binary > 0)

{

rem = binary % 10000;

switch(rem)

{

case 0:

strcat(hex, "0");

break;

case 1:

strcat(hex, "1");

break;

case 10:

strcat(hex, "2");

break;

case 11:

strcat(hex, "3");

break;

case 100:

strcat(hex, "4");

break;

case 101:

strcat(hex, "5");

break;

case 110:

strcat(hex, "6");

break;

case 111:

strcat(hex, "7");

break;

case 1000:

strcat(hex, "8");

break;

case 1001:

strcat(hex, "9");

break;

case 1010:

strcat(hex, "A");

break;

case 1011:

strcat(hex, "B");

break;

case 1100:

strcat(hex, "C");

break;

case 1101:

strcat(hex, "D");

break;

case 1110:

strcat(hex, "E");

break;

case 1111:

strcat(hex, "F");

break;

}

binary /= 10000;

}

strrev(hex);

printf("Octal number: %lld\n", octal);

printf("Hexadecimal number: %s", hex);

return 0;

}

/\*\*

\* C program to count frequency of digits in a given number

\*/

#include <stdio.h>

#define BASE 10 /\* Constant \*/

int main()

{

long long num, n;

int i, lastDigit;

int freq[BASE];

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%lld", &num);

/\* Initialize frequency array with 0 \*/

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

{

freq[i] = 0;

}

/\* Copy the value of 'num' to 'n' \*/

n = num;

/\* Run till 'n' is not equal to zero \*/

while(n != 0)

{

/\* Get last digit \*/

lastDigit = n % 10;

/\* Remove last digit \*/

n /= 10;

/\* Increment frequency array \*/

freq[lastDigit]++;

}

/\* Print frequency of each digit \*/

printf("Frequency of each digit in %lld is: \n", num);

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

{

printf("Frequency of %d = %d\n", i, freq[i]);

}

return 0;

}

/\*\*

\* C program to count minimum number of notes in an amount

\*/

#include <stdio.h>

int main()

{

int amount;

int note500, note100, note50, note20, note10, note5, note2, note1;

/\* Initialize all notes to 0 \*/

note500 = note100 = note50 = note20 = note10 = note5 = note2 = note1 = 0;

/\* Input amount from user \*/

printf("Enter amount: ");

scanf("%d", &amount);

if(amount >= 500)

{

note500 = amount/500;

amount -= note500 \* 500;

}

if(amount >= 100)

{

note100 = amount/100;

amount -= note100 \* 100;

}

if(amount >= 50)

{

note50 = amount/50;

amount -= note50 \* 50;

}

if(amount >= 20)

{

note20 = amount/20;

amount -= note20 \* 20;

}

if(amount >= 10)

{

note10 = amount/10;

amount -= note10 \* 10;

}

if(amount >= 5)

{

note5 = amount/5;

amount -= note5 \* 5;

}

if(amount >= 2)

{

note2 = amount /2;

amount -= note2 \* 2;

}

if(amount >= 1)

{

note1 = amount;

}

/\* Print required notes \*/

printf("Total number of notes = \n");

printf("500 = %d\n", note500);

printf("100 = %d\n", note100);

printf("50 = %d\n", note50);

printf("20 = %d\n", note20);

printf("10 = %d\n", note10);

printf("5 = %d\n", note5);

printf("2 = %d\n", note2);

printf("1 = %d\n", note1);

return 0;

}

/\*\*

\* C program to count minimum number of notes in an amount

\*/

#include <stdio.h>

int main()

{

int amount;

int note500, note100, note50, note20, note10, note5, note2, note1;

/\* Initialize all notes to 0 \*/

note500 = note100 = note50 = note20 = note10 = note5 = note2 = note1 = 0;

/\* Input amount from user \*/

printf("Enter amount: ");

scanf("%d", &amount);

if(amount >= 500)

{

note500 = amount/500;

amount -= note500 \* 500;

}

if(amount >= 100)

{

note100 = amount/100;

amount -= note100 \* 100;

}

if(amount >= 50)

{

note50 = amount/50;

amount -= note50 \* 50;

}

if(amount >= 20)

{

note20 = amount/20;

amount -= note20 \* 20;

}

if(amount >= 10)

{

note10 = amount/10;

amount -= note10 \* 10;

}

if(amount >= 5)

{

note5 = amount/5;

amount -= note5 \* 5;

}

if(amount >= 2)

{

note2 = amount /2;

amount -= note2 \* 2;

}

if(amount >= 1)

{

note1 = amount;

}

/\* Print required notes \*/

printf("Total number of notes = \n");

printf("500 = %d\n", note500);

printf("100 = %d\n", note100);

printf("50 = %d\n", note50);

printf("20 = %d\n", note20);

printf("10 = %d\n", note10);

printf("5 = %d\n", note5);

printf("2 = %d\n", note2);

printf("1 = %d\n", note1);

return 0;

}

/\*\*

\* C program to count number of digits in an integer

\*/

#include <stdio.h>

int main()

{

long long num;

int count = 0;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%lld", &num);

/\* Run loop till num is greater than 0 \*/

do

{

/\* Increment digit count \*/

count++;

/\* Remove last digit of 'num' \*/

num /= 10;

} while(num != 0);

printf("Total digits: %d", count);

return 0;

}

/\*\*

\* C program to display all even numbers from 1 to n without if

\*/

#include <stdio.h>

int main()

{

int i, n;

/\* Input upper limit of even number from user \*/

printf("Print all even numbers till: ");

scanf("%d", &n);

printf("All even numbers from 1 to %d are: \n", n);

/\*

\* Start loop from 2 and increment by 2,

\* in each repetition

\*/

for(i=2; i<=n; i+=2)

{

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

}

return 0;

}

/\*\*

\* C program to enter marks of five subjects and find percentage and grade

\*/

#include <stdio.h>

int main()

{

int phy, chem, bio, math, comp;

float per;

/\* Input marks of five subjects from user \*/

printf("Enter five subjects marks: ");

scanf("%d%d%d%d%d", &phy, &chem, &bio, &math, &comp);

/\* Calculate percentage \*/

per = (phy + chem + bio + math + comp) / 5.0;

printf("Percentage = %.2f\n", per);

/\* Find grade according to the percentage \*/

if(per >= 90)

{

printf("Grade A");

}

else if(per >= 80)

{

printf("Grade B");

}

else if(per >= 70)

{

printf("Grade C");

}

else if(per >= 60)

{

printf("Grade D");

}

else if(per >= 40)

{

printf("Grade E");

}

else

{

printf("Grade F");

}

return 0;

}

/\*\*

\* C program to find 1s complement of a binary number

\*/

#include <stdio.h>

#define SIZE 8

int main()

{

char binary[SIZE + 1], onesComp[SIZE + 1];

int i, error=0;

printf("Enter %d bit binary value: ", SIZE);

/\* Input binary string from user \*/

gets(binary);

/\* Store all inverted bits of binary value to onesComp \*/

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

{

if(binary[i] == '1')

{

onesComp[i] = '0';

}

else if(binary[i] == '0')

{

onesComp[i] = '1';

}

else

{

printf("Invalid Input");

error = 1;

/\* Exits from loop \*/

break;

}

}

/\* Marks the end of onesComp string \*/

onesComp[SIZE] = '\0';

/\* Check if there are binary string contains no error \*/

if(error == 0)

{

printf("Original binary = %s\n", binary);

printf("Ones complement = %s", onesComp);

}

return 0;

}

/\*\*

\* C program to find all prime factors of a given number

\*/

#include <stdio.h>

int main()

{

int i, j, num, isPrime;

/\* Input a number from user \*/

printf("Enter any number to print Prime factors: ");

scanf("%d", &num);

printf("All Prime Factors of %d are: \n", num);

/\* Find all Prime factors \*/

for(i=2; i<=num; i++)

{

/\* Check 'i' for factor of num \*/

if(num%i==0)

{

/\* Check 'i' for Prime \*/

isPrime = 1;

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

{

if(i%j==0)

{

isPrime = 0;

break;

}

}

/\* If 'i' is Prime number and factor of num \*/

if(isPrime==1)

{

printf("%d, ", i);

}

}

}

return 0;

}

/\*\*

\* C program to find all roots of a quadratic equation

\*/

#include <stdio.h>

#include <math.h> /\* Used for sqrt() \*/

int main()

{

float a, b, c;

float root1, root2, imaginary;

float discriminant;

printf("Enter values of a, b, c of quadratic equation (aX^2 + bX + c): ");

scanf("%f%f%f", &a, &b, &c);

/\* Find discriminant of the equation \*/

discriminant = (b \* b) - (4 \* a \* c);

/\* Find the nature of discriminant \*/

if(discriminant > 0)

{

root1 = (-b + sqrt(discriminant)) / (2\*a);

root2 = (-b - sqrt(discriminant)) / (2\*a);

printf("Two distinct and real roots exists: %.2f and %.2f", root1, root2);

}

else if(discriminant == 0)

{

root1 = root2 = -b / (2 \* a);

printf("Two equal and real roots exists: %.2f and %.2f", root1, root2);

}

else if(discriminant < 0)

{

root1 = root2 = -b / (2 \* a);

imaginary = sqrt(-discriminant) / (2 \* a);

printf("Two distinct complex roots exists: %.2f + i%.2f and %.2f - i%.2f",

root1, imaginary, root2, imaginary);

}

return 0;

}

/\* To find cube of any number\*/

#include <stdio.h>

/\* Function declaration \*/

double cube(double num);

int main()

{

int num;

double c;

/\* Input number to find cube from user \*/

printf("Enter any number: ");

scanf("%d", &num);

c = cube(num);

printf("Cube of %d is %.2f", num, c);

return 0;

}

/\*\*

\* Function to find cube of any number

\*/

double cube(double num)

{

return (num \* num \* num);

}

/\*Program to find diameter, circumference and area of circle using functions\*/

#include <stdio.h>

#include <math.h> // Used for constant PI referred as M\_PI

/\* Function declaration \*/

double getDiameter(double radius);

double getCircumference(double radius);

double getArea(double radius);

int main()

{

float radius, dia, circ, area;

/\* Input radius of circle from user \*/

printf("Enter radius of circle: ");

scanf("%f", &radius);

dia = getDiameter(radius); // Call getDiameter function

circ = getCircumference(radius); // Call getCircumference function

area = getArea(radius); // Call getArea function

printf("Diameter of the circle = %.2f units\n", dia);

printf("Circumference of the circle = %.2f units\n", circ);

printf("Area of the circle = %.2f sq. units", area);

return 0;

}

/\*\*

\* Calculate diameter of circle whose radius is given

\*/

double getDiameter(double radius)

{

return (2 \* radius);

}

/\*\*

\* Calculate circumference of circle whose radius is given

\*/

double getCircumference(double radius)

{

return (2 \* M\_PI \* radius); // M\_PI = PI = 3.14 ...

}

/\*\*

\* Find area of circle whose radius is given

\*/

double getArea(double radius)

{

return (M\_PI \* radius \* radius); // M\_PI = PI = 3.14 ...

}

/\*\*

\* C program to find factorial of any number using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

unsigned long long fact(int num);

int main()

{

int num;

unsigned long long factorial;

/\* Input an integer from user \*/

printf("Enter any number: ");

scanf("%d", &num);

factorial = fact(num); // Call factorial function

printf("Factorial of %d is %llu", num, factorial);

return 0;

}

/\*\*

\* Function to compute and return factorial of any number recursively.

\*/

unsigned long long fact(int num)

{

// Base condition

if(num == 0)

return 1;

else

return num \* fact(num - 1);

}

/\*\*

\* C program to find GCD (HCF) of two numbers using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

int gcd(int a, int b);

int main()

{

int num1, num2, hcf;

/\* Input two numbers from user \*/

printf("Enter any two numbers to find GCD: ");

scanf("%d%d", &num1, &num2);

hcf = gcd(num1, num2);

printf("GCD of %d and %d = %d", num1, num2, hcf);

return 0;

}

/\*\*

\* Recursive approach of euclidean algorithm to find GCD of two numbers

\*/

int gcd(int a, int b)

{

if(b == 0)

return a;

else

return gcd(b, a%b);

}

/\*\*

\* C program to find HCF of two numbers

\*/

#include <stdio.h>

int main()

{

int i, num1, num2, min, hcf=1;

/\* Input two numbers from user \*/

printf("Enter any two numbers to find HCF: ");

scanf("%d%d", &num1, &num2);

/\* Find minimum between two numbers \*/

min = (num1<num2) ? num1 : num2;

for(i=1; i<=min; i++)

{

/\* If i is factor of both number \*/

if(num1%i==0 && num2%i==0)

{

hcf = i;

}

}

printf("HCF of %d and %d = %d\n", num1, num2, hcf);

return 0;

}

/\*\*

\* C program to find last digit of a number

\*/

#include <stdio.h>

int main()

{

int n, lastDigit;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &n);

/\* Get the last digit \*/

lastDigit = n % 10;

printf("Last digit = %d", lastDigit);

return 0;

}

/\*\*

\* C program to find LCM of any two numbers

\*/

#include <stdio.h>

int main()

{

int i, num1, num2, max, lcm=1;

/\* Input two numbers from user \*/

printf("Enter any two numbers to find LCM: ");

scanf("%d%d", &num1, &num2);

/\* Find maximum between num1 and num2 \*/

max = (num1 > num2) ? num1 : num2;

/\* First multiple to be checked \*/

i = max;

/\* Run loop indefinitely till LCM is found \*/

while(1)

{

if(i%num1==0 && i%num2==0)

{

/\*

\* If 'i' divides both 'num1' and 'num2'

\* then 'i' is the LCM.

\*/

lcm = i;

/\* Terminate the loop after LCM is found \*/

break;

}

/\*

\* If LCM is not found then generate next

\* multiple of max between both numbers

\*/

i += max;

}

printf("LCM of %d and %d = %d", num1, num2, lcm);

return 0;

}

/\*\*

\* C program to find LCM of two numbers using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

int lcm(int a, int b);

int main()

{

int num1, num2, LCM;

/\* Input two numbers from user \*/

printf("Enter any two numbers to find lcm: ");

scanf("%d%d", &num1, &num2);

/\*

\* Ensures that first parameter of LCM function

\* is always less than second

\*/

if(num1 > num2)

LCM = lcm(num2, num1);

else

LCM = lcm(num1, num2);

printf("LCM of %d and %d = %d", num1, num2, LCM);

return 0;

}

/\*\*

\* Recursive function to find lcm of two numbers 'a' and 'b'.

\* Here 'a' needs to be always less than 'b'.

\*/

int lcm(int a, int b)

{

static int multiple = 0;

/\* Increments multiple by adding max value to it \*/

multiple += b;

/\*

\* Base condition of recursion

\* If found a common multiple then return the multiple.

\*/

if((multiple % a == 0) && (multiple % b == 0))

{

return multiple;

}

else

{

return lcm(a, b);

}

}

/\* Program to find maximum and minimum between two numbers using functions\*/

#include <stdio.h>

/\* Function declarations \*/

int max(int num1, int num2);

int min(int num1, int num2);

int main()

{

int num1, num2, maximum, minimum;

/\* Input two numbers from user \*/

printf("Enter any two numbers: ");

scanf("%d%d", &num1, &num2);

maximum = max(num1, num2); // Call maximum function

minimum = min(num1, num2); // Call minimum function

printf("\nMaximum = %d\n", maximum);

printf("Minimum = %d", minimum);

return 0;

}

/\*\*

\* Find maximum between two numbers.

\*/

int max(int num1, int num2)

{

return (num1 > num2 ) ? num1 : num2;

}

/\*\*

\* Find minimum between two numbers.

\*/

int min(int num1, int num2)

{

return (num1 > num2 ) ? num2 : num1;

}

Output

/\*\*

\* C program to find maximum between three numbers using nested if

\*/

#include <stdio.h>

int main()

{

int num1, num2, num3, max;

/\* Input three numbers from user \*/

printf("Enter three numbers: ");

scanf("%d%d%d", &num1, &num2, &num3);

if(num1 > num2)

{

if(num1 > num3)

{

/\* If num1 > num2 and num1 > num3 \*/

max = num1;

}

else

{

/\* If num1 > num2 but num1 > num3 is not true \*/

max = num3;

}

}

else

{

if(num2 > num3)

{

/\* If num1 is not > num2 and num2 > num3 \*/

max = num2;

}

else

{

/\* If num1 is not > num2 and num2 > num3 \*/

max = num3;

}

}

/\* Print maximum value \*/

printf("Maximum among all three numbers = %d", max);

return 0;

}

/\*\*

\* C program to find maximum between two numbers

\*/

#include <stdio.h>

int main()

{

int num1, num2;

/\* Input two numbers from user \*/

printf("Enter two numbers: ");

scanf("%d%d", &num1, &num2);

/\* If num1 is maximum \*/

if(num1 > num2)

{

printf("%d is maximum", num1);

}

/\* If num2 is maximum \*/

if(num2 > num1)

{

printf("%d is maximum", num2);

}

/\* Additional condition check for equality \*/

if(num1 == num2)

{

printf("Both are equal");

}

return 0;

}

/\*\*

\* C program to find nth Fibonacci term using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

unsigned long long fibo(int num);

int main()

{

int num;

unsigned long long fibonacci;

/\* Input a number from user \*/

printf("Enter any number to find nth fiboacci term: ");

scanf("%d", &num);

fibonacci = fibo(num);

printf("%d fibonacci term is %llu", num, fibonacci);

return 0;

}

/\*\*

\* Recursive function to find nth Fibonacci term

\*/

unsigned long long fibo(int num)

{

if(num == 0) //Base condition

return 0;

else if(num == 1) //Base condition

return 1;

else

return fibo(num-1) + fibo(num-2);

}

/\*\*

\* C program to find power of any number using for loop

\*/

#include <stdio.h>

int main()

{

int base, exponent;

long long power = 1;

int i;

/\* Input base and exponent from user \*/

printf("Enter base: ");

scanf("%d", &base);

printf("Enter exponent: ");

scanf("%d", &exponent);

/\* Multiply base, exponent times\*/

for(i=1; i<=exponent; i++)

{

power = power \* base;

}

printf("%d ^ %d = %lld", base, exponent, power);

return 0;

}

#include <stdio.h>

/\* Power function declaration \*/

double pow(double base, int expo);

int main()

{

double base, power;

int expo;

/\* Input base and exponent from user \*/

printf("Enter base: ");

scanf("%lf", &base);

printf("Enter exponent: ");

scanf("%d", &expo);

// Call pow function

power = pow(base, expo);

printf("%.2lf ^ %d = %f", base, expo, power);

return 0;

}

/\*\*

\* Calculate power of any number.

\* Returns base ^ expo

\*/

double pow(double base, int expo)

{

/\* Base condition \*/

if(expo == 0)

return 1;

else if(expo > 0)

return base \* pow(base, expo - 1);

else

return 1 / pow(base, -expo);

}

/\*\*

\* C program to find reverse of a number

\*/

#include <stdio.h>

int main()

{

int num, reverse = 0;

/\* Input a number from user \*/

printf("Enter any number to find reverse: ");

scanf("%d", &num);

/\* Repeat the till 'num' becomes 0 \*/

while(num != 0)

{

/\*

\* Increase place value of reverse and

\* add last digit to reverse

\*/

reverse = (reverse \* 10) + (num % 10);

/\* Remove last digit from 'num' \*/

num /= 10;

}

printf("Reverse = %d", reverse);

return 0;

}

/\*\*

\* C program to find reverse of any number using recursion

\*/

#include <stdio.h>

#include <math.h>

/\* Fuction declaration \*/

int reverse(int num);

int main()

{

int num, rev;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &num);

/\* Call the function to reverse number \*/

rev = reverse(num);

printf("Reverse of %d = %d", num, rev);

return 0;

}

/\*\*

\* Recursive function to find reverse of any number

\*/

int reverse(int num)

{

// Find total digits in num

int digit = (int) log10(num);

// Base condition

if(num == 0)

return 0;

return ((num%10 \* pow(10, digit)) + reverse(num/10));

}

/\*\*

\* C program to find sum of all even or odd numbers in given range using recursion

\*/

#include <stdio.h>

int sumOfEvenOdd(int start, int end);

int main()

{

int start, end, sum;

/\* Input lower and upper limit from user \*/

printf("Enter lower limit: ");

scanf("%d", &start);

printf("Enter upper limit: ");

scanf("%d", &end);

printf("Sum of even/odd numbers between %d to %d = %d\n", start, end, sumOfEvenOdd(start, end));

return 0;

}

/\*\*

\* Find sum of all even or odd numbers recursively.

\*/

int sumOfEvenOdd(int start, int end)

{

/\* Base condition \*/

if(start > end)

return 0;

else

return (start + sumOfEvenOdd(start + 2, end));

}

/\*\*

\* C program to find sum of first and last digit of a number using loop

\*/

#include <stdio.h>

int main()

{

int num, sum=0, firstDigit, lastDigit;

/\* Input a number from user \*/

printf("Enter any number to find sum of first and last digit: ");

scanf("%d", &num);

/\* Find last digit to sum \*/

lastDigit = num % 10;

/\* Copy num to first digit \*/

firstDigit = num;

/\* Find the first digit by dividing num by 10 until first digit is left \*/

while(num >= 10)

{

num = num / 10;

}

firstDigit = num;

/\* Find sum of first and last digit\*/

sum = firstDigit + lastDigit;

printf("Sum of first and last digit = %d", sum);

return 0;

}

\*\*

\* C program to find sum of its digits of a number

\*/

#include <stdio.h>

int main()

{

int num, sum=0;

/\* Input a number from user \*/

printf("Enter any number to find sum of its digit: ");

scanf("%d", &num);

/\* Repeat till num becomes 0 \*/

while(num!=0)

{

/\* Find last digit of num and add to sum \*/

sum += num % 10;

/\* Remove last digit from num \*/

num = num / 10;

}

printf("Sum of digits = %d", sum);

return 0;

}

/\*\*

\* C program to find sum of natural numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, n, sum=0;

/\* Input upper limit from user \*/

printf("Enter upper limit: ");

scanf("%d", &n);

/\* Find sum of all numbers \*/

for(i=1; i<=n; i++)

{

sum += i;

}

printf("Sum of first %d natural numbers = %d", n, sum);

return 0;

}

/\*\*

\* C program to find sum of natural numbers from 1 to n using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

int sumOfNaturalNumbers(int start, int end);

int main()

{

int start, end, sum;

/\* Input lower and upper limit from user \*/

printf("Enter lower limit: ");

scanf("%d", &start);

printf("Enter upper limit: ");

scanf("%d", &end);

sum = sumOfNaturalNumbers(start, end);

printf("Sum of natural numbers from %d to %d = %d", start, end, sum);

return 0;

}

/\*\*

\* Recursively find the sum of natural number

\*/

int sumOfNaturalNumbers(int start, int end)

{

if(start == end)

return start;

else

return start + sumOfNaturalNumbers(start + 1, end);

}

/\*\*

\* C program to find sum of prime numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, j, end, isPrime, sum=0;

/\* Input upper limit from user \*/

printf("Find sum of all prime between 1 to : ");

scanf("%d", &end);

/\* Find all prime numbers between 1 to end \*/

for(i=2; i<=end; i++)

{

/\* Check if the current number i is Prime or not \*/

isPrime = 1;

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

{

if(i%j==0)

{

/\* 'i' is not prime \*/

isPrime = 0;

break;

}

}

/\*

\* If 'i' is Prime then add to sum

\*/

if(isPrime==1)

{

sum += i;

}

}

printf("Sum of all prime numbers between 1 to %d = %d", end, sum);

return 0;

}

/\*\*

\* C program to find twos complement of a binary number

\*/

#include <stdio.h>

#define SIZE 8

int main()

{

char binary[SIZE + 1], onesComp[SIZE + 1], twosComp[SIZE + 1];

int i, carry=1;

printf("Enter %d bit binary value: ", SIZE);

/\* Input 8-bit binary string \*/

gets(binary);

/\* Find ones complement of the binary number \*/

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

{

if(binary[i] == '1')

{

onesComp[i] = '0';

}

else if(binary[i] == '0')

{

onesComp[i] = '1';

}

}

onesComp[SIZE] = '\0';

/\*

\* Add 1 to the ones complement

\*/

for(i=SIZE-1; i>=0; i--)

{

if(onesComp[i] == '1' && carry == 1)

{

twosComp[i] = '0';

}

else if(onesComp[i] == '0' && carry == 1)

{

twosComp[i] = '1';

carry = 0;

}

else

{

twosComp[i] = onesComp[i];

}

}

twosComp[SIZE] = '\0';

printf("Original binary = %s\n", binary);

printf("Ones complement = %s\n", onesComp);

printf("Twos complement = %s\n", twosComp);

return 0;

}

/\*\*

\* C program to list all prime number between an interval using function.

\*/

#include <stdio.h>

/\* Function declarations \*/

int isPrime(int num);

void printPrimes(int lowerLimit, int upperLimit);

int main()

{

int lowerLimit, upperLimit;

printf("Enter the lower and upper limit to list primes: ");

scanf("%d%d", &lowerLimit, &upperLimit);

// Call function to print all primes between the given range.

printPrimes(lowerLimit, upperLimit);

return 0;

}

/\*\*

\* Print all prime numbers between lower limit and upper limit.

\*/

void printPrimes(int lowerLimit, int upperLimit)

{

printf("All prime number between %d to %d are: ", lowerLimit, upperLimit);

while(lowerLimit <= upperLimit)

{

// Print if current number is prime.

if(isPrime(lowerLimit))

{

printf("%d, ", lowerLimit);

}

lowerLimit++;

}

}

/\*\*

\* Check whether a number is prime or not.

\* Returns 1 if the number is prime otherwise 0.

\*/

int isPrime(int num)

{

int i;

for(i=2; i<=num/2; i++)

{

/\*

\* If the number is divisible by any number

\* other than 1 and self then it is not prime

\*/

if(num % i == 0)

{

return 0;

}

}

return 1;

}

/\*\*

\* C program to print all alphabets from a to z

\*/

#include <stdio.h>

int main()

{

char ch;

printf("Alphabets from a - z are: \n");

for(ch='a'; ch<='z'; ch++)

{

printf("%c\n", ch);

}

return 0;

}

/\*\*

\* C program to print all Armstrong numbers between a given range

\*/

#include <stdio.h>

/\* Function declarations \*/

int isArmstrong(int num);

void printArmstrong(int start, int end);

int main()

{

int start, end;

/\* Input lower and upper limit to of armstrong numbers \*/

printf("Enter lower limit to print armstrong numbers: ");

scanf("%d", &start);

printf("Enter upper limit to print armstrong numbers: ");

scanf("%d", &end);

printf("All armstrong numbers between %d to %d are: \n", start, end);

printArmstrong(start, end);

return 0;

}

/\*\*

\* Check whether the given number is armstrong number or not.

\* Returns 1 if the number is armstrong otherwise 0.

\*/

int isArmstrong(int num)

{

int temp, lastDigit, sum;

temp = num;

sum = 0;

/\* Calculate sum of cube of digits \*/

while(temp != 0)

{

lastDigit = temp % 10;

sum += lastDigit \* lastDigit \* lastDigit;

temp /= 10;

}

/\*

\* Check if sum of cube of digits equals

\* to original number.

\*/

if(num == sum)

return 1;

else

return 0;

}

/\*\*

\* Print all armstrong numbers between start and end.

\*/

void printArmstrong(int start, int end)

{

/\*

\* Iterates from start to end and print the current number

\* if it is armstrong

\*/

while(start <= end)

{

if(isArmstrong(start))

{

printf("%d, ", start);

}

start++;

}

}

Output

/\*C program to print all Armstrong Numbers from 1 to N. \*/

#include <stdio.h>

/\*function to check Armstrong Number \*/

int checkArmstrong(int num){

int tempNumber,rem,sum;

tempNumber=num;

sum=0;

while(tempNumber!=0)

{

rem=tempNumber%10;

sum=sum + (rem\*rem\*rem);

tempNumber/=10;

}

/\* checking number is Armstrong or not \*/

if(sum==num)

return 1;

else

return 0;

}

int main()

{

int i,n;

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

scanf("%d",&n);

printf("All Armstrong numbers from 1 to %d:\n",n);

for(i=1;i<=n;i++)

{

if(checkArmstrong(i))

printf("%d,",i);

}

return 0;

}

Enter

/\*\*

\* C program to print all factors of a number

\*/

#include <stdio.h>

int main()

{

int i, num;

/\* Input number from user \*/

printf("Enter any number to find its factor: ");

scanf("%d", &num);

printf("All factors of %d are: \n", num);

/\* Iterate from 1 to num \*/

for(i=1; i<=num; i++)

{

/\*

\* If num is exactly divisible by i

\* Then i is a factor of num

\*/

if(num % i == 0)

{

printf("%d, ",i);

}

}

return 0;

}

/\*\*

\* C program to print all natural numbers from 1 to n using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

void printNaturalNumbers(int lowerLimit, int upperLimit);

int main()

{

int lowerLimit, upperLimit;

/\* Input lower and upper limit from user \*/

printf("Enter lower limit: ");

scanf("%d", &lowerLimit);

printf("Enter upper limit: ");

scanf("%d", &upperLimit);

printf("All natural numbers from %d to %d are: ", lowerLimit, upperLimit);

printNaturalNumbers(lowerLimit, upperLimit);

return 0;

}

/\*\*

\* Recursively prints all natural number between the given range.

\*/

void printNaturalNumbers(int lowerLimit, int upperLimit)

{

if(lowerLimit > upperLimit)

return;

printf("%d, ", lowerLimit);

// Recursively call the function to print next number

printNaturalNumbers(lowerLimit + 1, upperLimit);

}

/\*\*

\* C program to print all natural numbers from 1 to n

\*/

#include <stdio.h>

int main()

{

int i, n;

/\* Input upper limit from user \*/

printf("Enter any number: ");

scanf("%d", &n);

printf("Natural numbers from 1 to %d : \n", n);

/\*

\* Start loop counter from 1 (i=1) and go till n (i<=n)

\* increment the loop count by 1 to get the next value.

\* For each repetition print the value of i.

\*/

for(i=1; i<=n; i++)

{

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

}

return 0;

}

/\*\*

\* C program to print all Odd numbers from 1 to n

\*/

#include <stdio.h>

int main()

{

int i, n;

/\* Input upper limit from user \*/

printf("Print odd numbers till: ");

scanf("%d", &n);

printf("All odd numbers from 1 to %d are: \n", n);

/\* Start loop from 1 and increment it by 1 \*/

for(i=1; i<=n; i++)

{

/\* If 'i' is odd then print it \*/

if(i%2!=0)

{

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

}

}

return 0;

}

/\*\*

\* C program to print all Perfect numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, j, end, sum;

/\* Input upper limit to print perfect number \*/

printf("Enter upper limit: ");

scanf("%d", &end);

printf("All Perfect numbers between 1 to %d:\n", end);

/\* Iterate from 1 to end \*/

for(i=1; i<=end; i++)

{

sum = 0;

/\* Check whether the current number i is Perfect number or not \*/

for(j=1; j<i; j++)

{

if(i % j == 0)

{

sum += j;

}

}

/\* If the current number i is Perfect number \*/

if(sum == i)

{

printf("%d, ", i);

}

}

/\*\*

\* C program to print all perfect numbers in given range using function

\*/

#include <stdio.h>

/\* Function declarations \*/

int isPerfect(int num);

void printPerfect(int start, int end);

int main()

{

int start, end;

/\* Input lower and upper limit to print perfect numbers \*/

printf("Enter lower limit to print perfect numbers: ");

scanf("%d", &start);

printf("Enter upper limit to print perfect numbers: ");

scanf("%d", &end);

printf("All perfect numbers between %d to %d are: \n", start, end);

printPerfect(start, end);

return 0;

}

/\*\*

\* Check whether the given number is perfect or not.

\* Returns 1 if the number is perfect otherwise 0.

\*/

int isPerfect(int num)

{

int i, sum;

/\* Finds sum of all proper divisors \*/

sum = 0;

for(i=1; i<num; i++)

{

if(num % i == 0)

{

sum += i;

}

}

/\*

\* If sum of proper positive divisors equals to given number

\* then the number is perfect number

\*/

if(sum == num)

return 1;

else

return 0;

}

/\*\*

\* Print all perfect numbers between given range start and end.

\*/

void printPerfect(int start, int end)

{

/\* Iterates from start to end \*/

while(start <= end)

{

if(isPerfect(start))

{

printf("%d, ", start);

}

start++;

}

}

/\*\*

\* C program to print all prime numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, j, end, isPrime; // isPrime is used as flag variable

/\* Input upper limit to print prime \*/

printf("Find prime numbers between 1 to : ");

scanf("%d", &end);

printf("All prime numbers between 1 to %d are:\n", end);

/\* Find all Prime numbers between 1 to end \*/

for(i=2; i<=end; i++)

{

/\* Assume that the current number is Prime \*/

isPrime = 1;

/\* Check if the current number i is prime or not \*/

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

{

/\*

\* If i is divisible by any number other than 1 and self

\* then it is not prime number

\*/

if(i%j==0)

{

isPrime = 0;

break;

}

}

/\* If the number is prime then print \*/

if(isPrime==1)

{

printf("%d, ", i);

}

}

return 0;

}

/\*\*

\* C program to print all Strong Numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, j, cur, lastDigit, end;

long long fact, sum;

/\* Input upper limit from user \*/

printf("Enter upper limit: ");

scanf("%d", &end);

printf("All Strong numbers between 1 to %d are:\n", end);

/\* Iterate from 1 to end \*/

for(i=1; i<=end; i++)

{

/\* Number to check for strong number \*/

cur = i;

sum = 0;

/\* Find the sum of factorial of digits \*/

while(cur > 0)

{

fact = 1ll;

lastDigit = cur % 10;

/\* Find factorial of last digit of current num. \*/

for( j=1; j<=lastDigit; j++)

{

fact = fact \* j;

}

sum += fact;

cur /= 10;

}

/\* Print 'i' if it is strong number \*/

if(sum == i)

{

printf("%d, ", i);

}

}

return 0;

}

/\*\*

\* C program to print ASCII values of all characters.

\*/

#include <stdio.h>

int main()

{

int i;

/\* Print ASCII values from 0 to 255 \*/

for(i=0; i<=255; i++)

{

printf("ASCII value of character %c = %d\n", i, i);

}

return 0;

}

/\*\*

\* C program to print box number pattern of 1's and 0's

\*/

#include <stdio.h>

int main()

{

int rows, cols, i, j;

/\* Input rows and columns from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

for(i=1; i<=rows; i++)

{

for(j=1; j<=cols; j++)

{

/\*

\* Print 1 if its first or last row

\* Print 1 if its first or last column

\*/

if(i==1 || i==rows || j==1 || j==cols)

{

printf("1");

}

else

{

printf("0");

}

}

printf("\n");

}

return 0;

}

/\*\*

\* C program to print box number pattern of 1's with 0 as center

\*/

#include <stdio.h>

int main()

{

int rows, cols, i, j;

int centerRow, centerCol;

/\* Input rows and columns from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

/\* Find center row and column \*/

centerRow = (rows + 1) / 2;

centerCol = (cols + 1) / 2;

for(i=1; i<=rows; i++)

{

for(j=1; j<=cols; j++)

{

if(centerCol == j && centerRow == i)

{

printf("0");

}

else if(cols%2 == 0 && centerCol+1 == j)

{

if(centerRow == i || (rows%2 == 0 && centerRow+1 == i))

printf("0");

else

printf("1");

}

else if(rows%2 == 0 && centerRow+1 == i)

{

if(centerCol == j || (cols%2 == 0 && centerCol+1 == j))

printf("0");

else

printf("1");

}

else

{

printf("1");

}

}

printf("\n");

}

return 0;

}

/\*\*

\* C program to print day name of week

\*/

#include <stdio.h>

int main()

{

int week;

/\* Input week number from user \*/

printf("Enter week number (1-7): ");

scanf("%d", &week);

if(week == 1)

{

printf("Monday");

}

else if(week == 2)

{

printf("Tuesday");

}

else if(week == 3)

{

printf("Wednesday");

}

else if(week == 4)

{

printf("Thursday");

}

else if(week == 5)

{

printf("Friday");

}

else if(week == 6)

{

printf("Saturday");

}

else if(week == 7)

{

printf("Sunday");

}

else

{

printf("Invalid Input! Please enter week number between 1-7.");

}

return 0;

}

/\*\*

\* C program to print equilateral triangle or pyramid star pattern

\*/

#include <stdio.h>

int main()

{

int i, j, rows;

/\* Input number of rows to print \*/

printf("Enter number of rows : ");

scanf("%d", &rows);

/\* Iterate through rows \*/

for(i=1; i<=rows; i++)

{

/\* Print leading spaces \*/

for(j=i; j<rows; j++)

{

printf(" ");

}

/\* Print star \*/

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

{

printf("\*");

}

/\* Move to next line \*/

printf("\n");

}

return 0;

}

/\*\*

\* C program to print even or odd numbers in given range using recursion

\*/

#include <stdio.h>

/\* Function declaration \*/

void printEvenOdd(int cur, int limit);

int main()

{

int lowerLimit, upperLimit;

// Input lower and upper limit from user

printf("Enter lower limit: ");

scanf("%d", &lowerLimit);

printf("Enter upper limit: ");

scanf("%d", &upperLimit);

printf("Even/odd Numbers from %d to %d are: ", lowerLimit, upperLimit);

printEvenOdd(lowerLimit, upperLimit);

return 0;

}

/\*\*

\* Recursive function to print even or odd numbers in a given range.

\*/

void printEvenOdd(int cur, int limit)

{

if(cur > limit)

return;

printf("%d, ", cur);

// Recursively call to printEvenOdd to get next value

printEvenOdd(cur + 2, limit);

}

/\*\*

\* C program to print Fibonacci series up to n terms

\*/

#include <stdio.h>

int main()

{

int a, b, c, i, terms;

/\* Input number from user \*/

printf("Enter number of terms: ");

scanf("%d", &terms);

/\* Fibonacci magic initialization \*/

a = 0;

b = 1;

c = 0;

printf("Fibonacci terms: \n");

/\* Iterate through n terms \*/

for(i=1; i<=terms; i++)

{

printf("%d, ", c);

a = b; // Copy n-1 to n-2

b = c; // Copy current to n-1

c = a + b; // New term

}

return 0;

}

/\*\*

\* C program to print half diamond star pattern series.

\*/

#include<stdio.h>

int main()

{

int i, j, N, columns;

/\* Input number of columns from user \*/

printf("Enter number of columns:");

scanf("%d",&N);

columns=1;

for(i=1;i<N\*2;i++)

{

for(j=1; j<=columns; j++)

{

printf("\*");

}

if(i < N)

{

/\* Increment number of columns per row for upper part \*/

columns++;

}

else

{

/\* Decrement number of columns per row for lower part \*/

columns--;

}

/\* Move to next line \*/

printf("\n");

}

return 0;

}

/\*\*

\* C program to print hollow inverted pyramid star pattern

\*/

#include <stdio.h>

int main()

{

int i, j, rows;

/\* Input rows to print from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

/\* Iterate through rows \*/

for(i=1; i<=rows; i++)

{

/\* Print leading spaces \*/

for(j=1; j<i; j++)

{

printf(" ");

}

/\* Print hollow pyramid \*/

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

{

/\*

\* Print star for first row(i==1),

\* ith column (j==1) and for

\* last column (rows\*2-(2\*i-1))

\*/

if(i==1 || j==1 || j==(rows\*2 - (2\*i - 1)))

{

printf("\*");

}

else

{

printf(" ");

}

}

/\* Move to next line \*/

printf("\n");

}

return 0;

}

/\*\*

\* C program to print hollow pyramid triangle star pattern

\*/

#include <stdio.h>

int main()

{

int i, j, rows;

/\* Input rows to print from user \*/

printf("Enter number of rows : ");

scanf("%d", &rows);

for(i=1; i<=rows; i++)

{

/\* Print trailing spaces \*/

for(j=i; j<rows; j++)

{

printf(" ");

}

/\* Print hollow pyramid \*/

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

{

/\*

\* Print star for last row (i==rows),

\* first column(j==1) and for

\* last column (j==(2\*i-1)).

\*/

if(i==rows || j==1 || j==(2\*i-1))

{

printf("\*");

}

else

{

printf(" ");

}

}

/\* Move to next line \*/

printf("\n");

}

return 0;

}

/\*\*

\* C program to print mirrored half diamond star pattern

\*/

#include <stdio.h>

int main()

{

int i, j, N;

int star, spaces;

/\* Input number of columns to print from user \*/

printf("Enter number of columns : ");

scanf("%d", &N);

spaces = N-1;

star = 1;

/\* Iterate through rows \*/

for(i=1; i<N\*2; i++)

{

/\* Print leading spaces \*/

for(j=1; j<=spaces; j++)

printf(" ");

/\* Print stars \*/

for(j=1; j<=star; j++)

printf("\*");

/\* Move to next line \*/

printf("\n");

if(i < N)

{

star++;

spaces--;

}

else

{

star--;

spaces++;

}

}

return 0;

}

/\*\*

\* C program to print multiplication table of a number

\*/

#include <stdio.h>

int main()

{

int i, num;

/\* Input a number to print table \*/

printf("Enter number to print table: ");

scanf("%d", &num);

for(i=1; i<=10; i++)

{

printf("%d \* %d = %d\n", num, i, (num\*i));

}

return 0;

}

/\*\*

\* C program to print number in words

\*/

#include <stdio.h>

int main()

{

int n, num = 0;

/\* Input number from user \*/

printf("Enter any number to print in words: ");

scanf("%d", &n);

/\* Store reverse of n in num \*/

while(n != 0)

{

num = (num \* 10) + (n % 10);

n /= 10;

}

/\*

\* Extract last digit of number and print corresponding digit in words

\* till num becomes 0

\*/

while(num != 0)

{

switch(num % 10)

{

case 0:

printf("Zero ");

break;

case 1:

printf("One ");

break;

case 2:

printf("Two ");

break;

case 3:

printf("Three ");

break;

case 4:

printf("Four ");

break;

case 5:

printf("Five ");

break;

case 6:

printf("Six ");

break;

case 7:

printf("Seven ");

break;

case 8:

printf("Eight ");

break;

case 9:

printf("Nine ");

break;

}

num = num / 10;

}

return 0;

}

/\*\*

\* C program to print number of days in a month

\*/

#include <stdio.h>

int main()

{

int month;

/\* Input month number from user \*/

printf("Enter month number (1-12): ");

scanf("%d", &month);

if(month == 1)

{

printf("31 days");

}

else if(month == 2)

{

printf("28 or 29 days");

}

else if(month == 3)

{

printf("31 days");

}

else if(month == 4)

{

printf("30 days");

}

else if(month == 5)

{

printf("31 days");

}

else if(month == 6)

{

printf("30 days");

}

else if(month == 7)

{

printf("31 days");

}

else if(month == 8)

{

printf("31 days");

}

else if(month == 9)

{

printf("30 days");

}

else if(month == 10)

{

printf("31 days");

}

else if(month == 11)

{

printf("30 days");

}

else if(month == 12)

{

printf("31 days");

}

else

{

printf("Invalid input! Please enter month number between (1-12).");

}

return 0;

}

/\*\*

\* C program to print number of days in a month

\*/

#include <stdio.h>

int main()

{

int month;

/\* Input month number from user \*/

printf("Enter month number (1-12): ");

scanf("%d", &month);

if(month == 1)

{

printf("31 days");

}

else if(month == 2)

{

printf("28 or 29 days");

}

else if(month == 3)

{

printf("31 days");

}

else if(month == 4)

{

printf("30 days");

}

else if(month == 5)

{

printf("31 days");

}

else if(month == 6)

{

printf("30 days");

}

else if(month == 7)

{

printf("31 days");

}

else if(month == 8)

{

printf("31 days");

}

else if(month == 9)

{

printf("30 days");

}

else if(month == 10)

{

printf("31 days");

}

else if(month == 11)

{

printf("30 days");

}

else if(month == 12)

{

printf("31 days");

}

else

{

printf("Invalid input! Please enter month number between (1-12).");

}

return 0;

}

/\*\*

\* C program to print number pattern of 1, 0 at even/odd rows

\*/

#include <stdio.h>

int main()

{

int rows, cols, i, j;

/\* Input rows and columns from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

for(i=1; i<=rows; i++)

{

for(j=1; j<=cols; j++)

{

// Print 1 if current row is odd

if(i%2 == 1)

{

printf("1");

}

else

{

printf("0");

}

}

printf("\n");

}

return 0;

}

/\*\*

\* C program to print number pattern with 1/0 at even/odd position

\*/

#include <stdio.h>

int main()

{

int rows, cols, i, j;

/\* Input rows and columns from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

for(i=1; i<=rows; i++)

{

for(j=1; j<=cols; j++)

{

// Print 1 if current column is even

if(j%2 == 1)

{

printf("0");

}

else

{

printf("1");

}

}

printf("\n");

}

return 0;

}

/\*\*

\* C program to print reverse pyramid star pattern

\*/

#include <stdio.h>

int main()

{

int i, j, rows;

/\* Input rows to print from user \*/

printf("Enter number of rows : ");

scanf("%d", &rows);

for(i=1; i<=rows; i++)

{

/\* Print leading spaces \*/

for(j=1; j<i; j++)

{

printf(" ");

}

/\* Print stars \*/

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

{

printf("\*");

}

/\* Move to next line \*/

printf("\n");

}

return 0;

}

/\*\*

\* C program to print square number pattern

\*/

#include <stdio.h>

int main()

{

int rows, cols, i, j;

/\* Input rows and columns from user \*/

printf("Enter number of rows: ");

scanf("%d", &rows);

printf("Enter number of columns: ");

scanf("%d", &cols);

/\* Iterate through rows \*/

for(i=1; i<=rows; i++)

{

/\* Iterate through columns \*/

for(j=1; j<=cols; j++)

{

printf("1");

}

printf("\n");

}

return 0;

}

/\*\*

\* C program to print strong numbers in a given interval using functions

\*/

#include <stdio.h>

/\* Function declaration \*/

long long fact(int num);

void printStrongNumbers(int start, int end);

int main()

{

int start, end;

/\* Input start and end range \*/

printf("Enter the lower limit to find strong number: ");

scanf("%d", &start);

printf("Enter the upper limit to find strong number: ");

scanf("%d", &end);

printf("All strong numbers between %d to %d are: \n", start, end);

printStrongNumbers(start, end);

return 0;

}

/\*\*

\* Print all strong numbers in a given range

\*/

void printStrongNumbers(int start, int end)

{

long long sum;

int num;

// Iterates from start to end

while(start != end)

{

sum = 0;

num = start;

// Calculate sum of factorial of digits

while(num != 0)

{

sum += fact(num % 10);

num /= 10;

}

// If sum of factorial of digits equal to current number

if(start == sum)

{

printf("%d, ", start);

}

start++;

}

}

/\*\*

\* Recursively find factorial of any number

\*/

long long fact(int num)

{

if(num == 0)

return 1;

else

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

}

/\*\*

\* C program to print sum of all even numbers between 1 to n

\*/

#include <stdio.h>

int main()

{

int i, n, sum=0;

/\* Input upper limit from user \*/

printf("Enter upper limit: ");

scanf("%d", &n);

for(i=2; i<=n; i+=2)

{

/\* Add current even number to sum \*/

sum += i;

}

printf("Sum of all even number between 1 to %d = %d", n, sum);

return 0;

}

/\*\*

\* C program to print the sum of all odd numbers from 1 to n

\*/

#include <stdio.h>

int main()

{

int i, n, sum=0;

/\* Input range to find sum of odd numbers \*/

printf("Enter upper limit: ");

scanf("%d", &n);

/\* Find the sum of all odd number \*/

for(i=1; i<=n; i+=2)

{

sum += i;

}

printf("Sum of odd numbers = %d", sum);

return 0;

}

/\*\*

\* C program to swap first and last digit of a number

\*/

#include <stdio.h>

#include <math.h>

int main()

{

int num, swappedNum;

int firstDigit, lastDigit, digits;

/\* Input number from user \*/

printf("Enter any number: ");

scanf("%d", &num);

/\* Find last digit \*/

lastDigit = num % 10;

/\* Find total number of digit - 1 \*/

digits = (int)log10(num);

/\* Find first digit \*/

firstDigit = (int)(num / pow(10, digits));

swappedNum = lastDigit;

swappedNum \*= (int) pow(10, digits);

swappedNum += num % ((int) pow(10, digits));

swappedNum -= lastDigit;

swappedNum += firstDigit;

printf("Original number = %d", num);

printf("Number after swapping first and last digit: %d", swappedNum);

return 0;

}

/\*\*

\* C program to whether a number is prime number or not

\*/

#include <stdio.h>

int main()

{

int i, num, isPrime;

/\*

\* isPrime is used as flag variable.

\* If isPrime = 0, then number is composite

\* else if isPrime = 1, then number is prime.

\* Initially I have assumed the number as prime.

\*/

isPrime = 1;

/\* Input a number from user \*/

printf("Enter any number to check prime: ");

scanf("%d", &num);

for(i=2; i<=num/2; i++)

{

/\* Check divisibility of num \*/

if(num%i==0)

{

/\* Set isPrime to 0 indicating it as composite number \*/

isPrime = 0;

/\* Terminate from loop \*/

break;

}

}

/\*

\* If isPrime contains 1 then it is prime

\*/

if(isPrime == 1 && num > 1)

{

printf("%d is prime number", num);

}

else

{

printf("%d is composite number", num);

}

return 0;

}

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**