**PF LAB NO. 9**

**Assignment Functions.**

**Name:** Ahmed Kasteer

**Roll Number:** 20F-0336

**Question 1:**

**By Val: Nth Term**

#include <iostream>

using namespace std;

int nthTerm(int N);

int main()

{

int a;

cout << "Input number to find if it belongs to the series." << endl;

cin >> a;

cout << nthTerm(a) << endl;

system("pause");

}

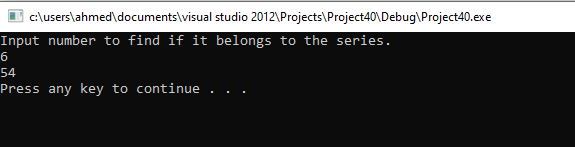
int nthTerm(int N)

{

int x = ( N \*((N/2) + ((N%2)\*2) + N));

return x;

}

****

**Question 2:**

**Quadrilateral:**

#include <iostream>

using namespace std;

bool collinear(int a1, int a2, int b1, int b2, int c1, int c2) {

if (0.5 \* (a1 \* (b2 - c2) + b1 \* (c2 - a2) + c1 \* (a2 - b2)) == 0) {

return 1;

}

return 0;

}

bool onSegment(int a1,int a2,int b1,int b2,int c1,int c2) {

if (((b2-a2)/(b1-a1))==((c2-a2)/(c1-a1))) {

return 1;

}

return 0;

}

int orientation(int a1,int a2, int b1,int b2,int c1,int c2) {

int x = 0;

if (collinear(a1,a2,b1,b2,c1,c2)==true) {

x = 3;

}

else if ((c2 - b2) / (c1 - b1)>0) {

x = 2;

}

else {

x = 1;

}

return x;

}

bool doIntersect(int a1,int a2,int b1,int b2,int c1,int c2,int d1,int d2) {

//(p1, q1, p2) and (p1, q1, q2) have different orientationsand –(p2, q2, p1) and (p2, q2, q1) have different orientations.

//general case:

if ((orientation(a1,a2,d1,d2,c1,c2)!=orientation(a1,a2,d1,d2,b1,b2)) && (orientation(c1,c2,b1,b2,a1,a2)!=orientation(c1,c2,b1,b2,d1,d2))) {

return 1;

}

//–(p1, q1, p2), (p1, q1, q2), (p2, q2, p1), and (p2, q2, q1) are all collinearand

//– the x - projections of(p1, q1) and (p2, q2) intersect

//– the y - projections of(p1, q1) and (p2, q2) intersect

//Special case:

if (collinear(a1,a2,b1,b2,c1,c2)==true && collinear(a1,a2,b1,b2,d1,d2)==true && collinear(c1,c2,d1,d2,a1,a2)==true && collinear(c1,c2,d1,d2,b1,b2)==true && onSegment(a1,a2,b1,b2,c1,c2)==true&&onSegment(c1,c2,d1,d2,b1,b2)==true) {

return 1;

}

return 0;

}

bool similar(int a1, int a2, int b1, int b2) {

if (a1 == b1 && a2 == b2) {

return 1;

}

return 0;

}

int no\_of\_quads(int a1, int a2, int b1, int b2, int c1, int c2, int d1, int d2) {

//check1:

if ((similar(a1,a2,b1,b2)==true)||(similar(a1,a2,c1,c2))||(similar(a1,a2,d1,d2))||(similar(b1,b2,c1,c2)) ||(similar(b1,b2,d1,d2))) {

return 0;

}

//check2:

if ((collinear(a1,a2,b1,b2,c1,c2)==true)|| (collinear(a1, a2, b1, b2, d1, d2) == true)|| (collinear(b1, b2, c1, c2, d1, d2) == true)|| (collinear(a1, a2, c1, c2, d1, d2) == true)) {

return 0;

}

int count = 0;

if (doIntersect(a1,a2,b1,b2,c1,c2,d1,d2)==true) {

return 1;

}

else {

return 3;

}

}

int main()

{

int p=1,q=9,r=-1,s=4,t=5,u=-1,v=5,w=9;

cout << no\_of\_quads(p, q, r, s, t, u, v, w);

system ("pause");

}

**Question 3:**

**Dice**

#include <iostream>

#include <cstdlib>

using namespace std;

int dice(int k);

int main()

{

int j = 0;

cout << dice(j);

system("pause");

}

int dice(int k)

{

int y, num1;

cout << "Enter number of turns to throw dice for." << endl;

cin >> y;

for (int i = 1 ; i <= y ; i++)

{

num1 = (rand() % 6) + 1;

k += num1 ;

}

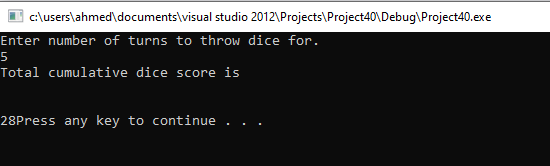
cout << "Total cumulative dice score is" << endl;

cout << endl;

cout << endl;

return k;

}

****

**Question 4:**

**Factorial Function**

#include <iostream>

using namespace std;

int factorial(int k);

int main()

{

int j = 0;

cout << factorial(j);

system("pause");

}

int factorial(int k)

{

int noFactorial = 1, num1;

cout << "Input number." << endl;

cin >> num1;

if ( num1 == 0 )

{

noFactorial = 1;

k = noFactorial;

return k;

}

else if( num1 > 0)

{

for (int i = 1 ; i <= num1 ; i++)

noFactorial \*= i;

k = noFactorial;

}

else

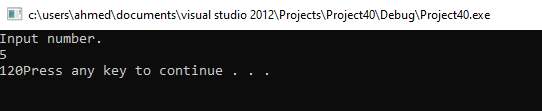
{

cout << "Number entered is less than 0" << endl;

}

return k;

}

****

**Question 5:**

#include <iostream>

using namespace std;

int larger(int a,int b) {

if (a>b) {

return a;

}

return b;

}

void largest(int n) {

if (n==0) {

cout << "Set is empty!" << endl;

}

else if (n==1) {

cout << "Enter:" << endl;

int x;

cin >> x;

cout <<"Largest:"<< x;

}

else if (n==2) {

cout << "Enter:" << endl;

int x;

int y;

cin >> x >> y;

cout<<"Largest:"<<larger(x,y);

}

else {

cout << "Enter:" << endl;

int max;

cin >> max;

int x = 0;

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

cin >> x;

if (x>max) {

max = x;

}

}

cout<<"Largest:"<< max;

}

}

int main()

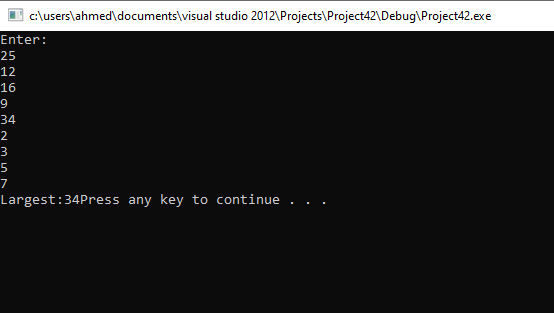
{

int q=9;

largest(q);

system ("pause");

}

****

**Question 6:**

#include <iostream>

using namespace std;

float CalculatePayment(int l,float r,int m,int t) {

return ((l \* (r / m)) / 1) - pow(1 + (r / m), -(m \* t));

}

float CalculateRemainingPayment(int l,float r,int m,int t,int k) {

return (CalculatePayment(l,r,m,t)\*(1-(pow(1+(r/m),-(m\*t-k)))));

}

void menu() {

int choice = 0;

while (choice!=2) {

cout << "1.Run" << endl << "2.Terminate" << endl;

cin >> choice;

if (choice==1) {

int l, m, t, k;

float r;

cout << "Enter Values for:" << endl;

cout << "-Loan Amount" << endl << "-Interest rate per year" << endl << "-Number of payments in a year" << endl << "-Loan time" << endl << "-Payments made" << endl;

cin >> l>>r>> m>> t>> k;

float R = 0;

float L= 0;

R=CalculatePayment(l,r,m,t);

L = CalculateRemainingPayment(l,r,m,t,k);

cout << "Periodic Payment:" << R << endl;

cout << "Unpaid Loan Balance:" << L << endl;

}

}

}

int main()

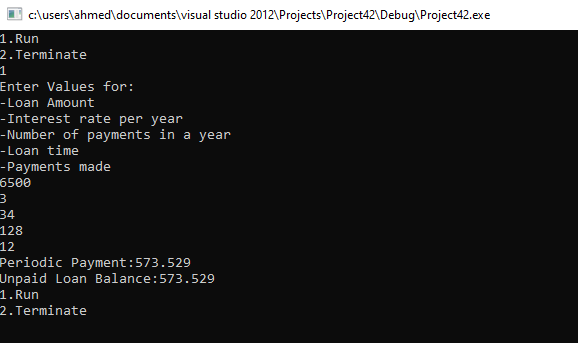
{

int q=9;

menu();

system ("pause");

}

****

**Question 7:**

#include <iostream>

using namespace std;

void perfectNums(int n) {

int sum = 0;

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

sum = 0;

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

if (i%j==0) {

sum = sum + j;

}

}

if (sum==i) {

cout << "Divisors:" << endl;

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

if (i % j == 0) {

cout << j << endl;

}

}

cout << "PerfNum:" << i << endl;

}

}

}

int main()

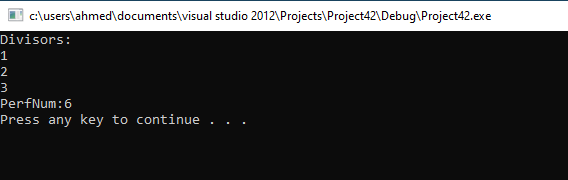
{

int q=8;

perfectNums(q);

system ("pause");

}

****