***Name - Akriti Choudhary***

***Roll number - 2005776***

***Lab5***

***Subject - OOP lab***

***Class - B14***

***Branch - CSE***

***Date- 26/08/2021***

***Question 1 :WAP to find volume of a sphere ,a cylinder and a cuboid using function overloading.***

#include <iostream>

#define pi 3.14

using namespace std;

class vol

{

public:

int volume(int r);

int volume(int r, int h);

int volume(int l, int b, int h);

};

int vol::volume(int r)

{

return 1.33 \* pi \* r \* r \* r;

}

int vol::volume(int r, int h)

{

return pi \* r \* r \* h;

}

int vol::volume(int l, int b, int h)

{

return l \* b \* h;

}

int main()

{

vol v;

int r, h, l, b;

int ch;

cout << "Enter choice:\n1 - volume of sphere\n2 - volume of cylinder\n3 - volume of cuboid";

cout << "\n-1 to exit\n";

cin >> ch;

while (ch != -1)

{

switch (ch)

{

case 1:

cout << "Enter the radius of sphere :" << endl;

cin >> r;

cout << "Result = " << v.volume(r)<<endl;

break;

case 2:

cout << "Enter the radius and height of cylinder :" << endl;

cin >> r >> h;

cout << "Result = " << v.volume(r, h)<<endl;

break;

case 3:

cout << "Enter the l, b and h of cuboid :" << endl;

cin >> l >> b >> h;

cout << "Result = " << v.volume(l, b, h)<<endl;

break;

default:

cout << "Wrong choice " << endl;

break;

}

cout << "Enter choice:\n1 - volume of sphere\n2 - volume of cylinder\n3 - volume of cuboid";

cout << "\n-1 to exit\n";

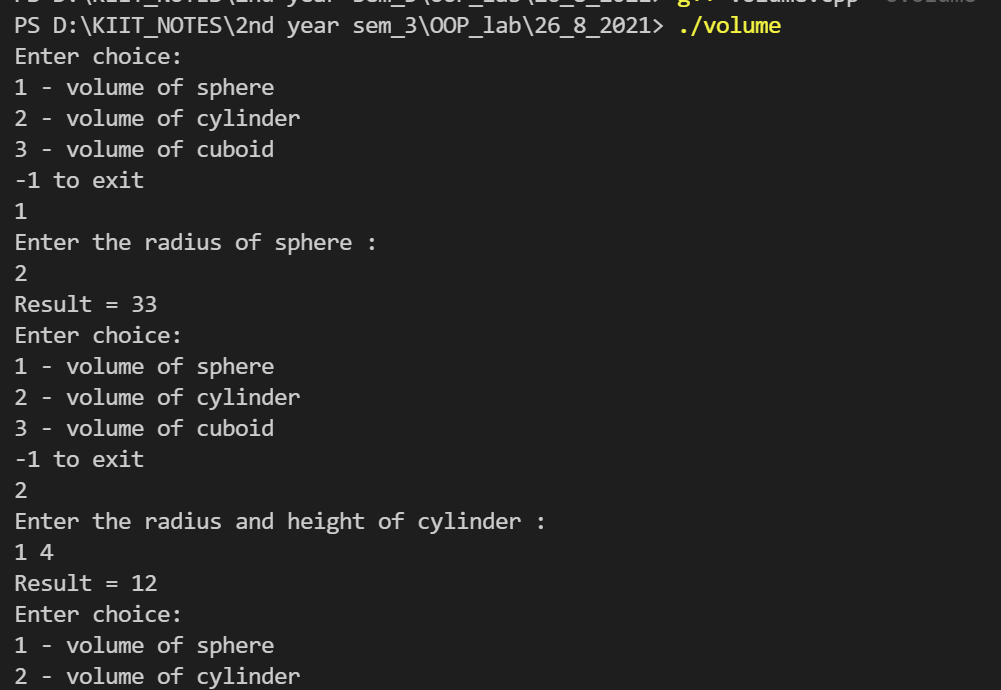
cin >> ch;

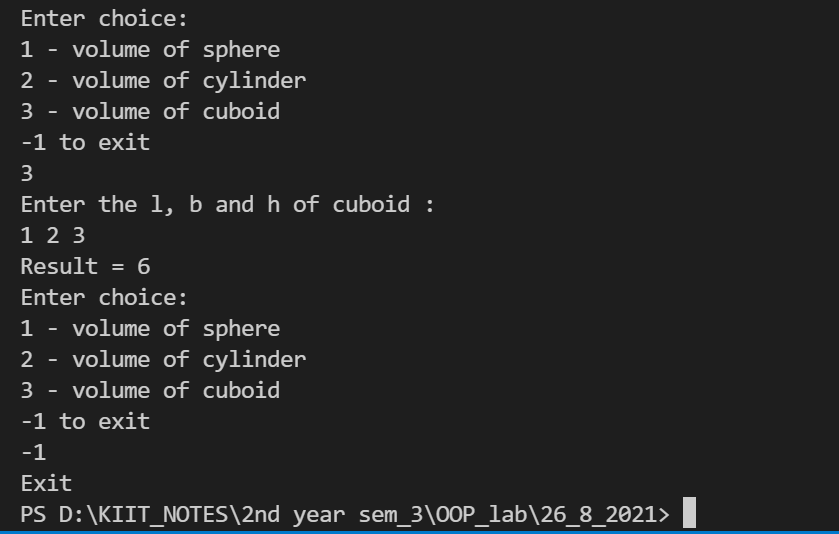
}

cout << "Exit" << endl;

return 0;

}





***Question 2 :WAP which displays a given character ,n number of times ,using a function .When the n value is not provided ,it should print the character 80 times .When both the character and n values is not provided,it should print ‘\*’ character 80 times.[use function overloading and default arguments]***

#include <iostream>

using namespace std;

class ch

{

public:

void display(char c = '\*', int n = 80);

};

void ch::display(char c, int n)

{

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

{

cout << c << " ";

if (i % 7 == 0)

cout << endl;

}

cout << endl;

}

int main()

{

char cha;

int num;

ch obj;

int n;

cout << "Printing 80 times using default argument with default value of n and ch:" << endl;

obj.display();

cout << "Enter the character :";

cin >> cha;

cout << "Printing using function overloading with default value of n :" << endl;

obj.display(cha);

cout << "Enter the value of n :";

cin >> n;

cout << "Enter the character :";

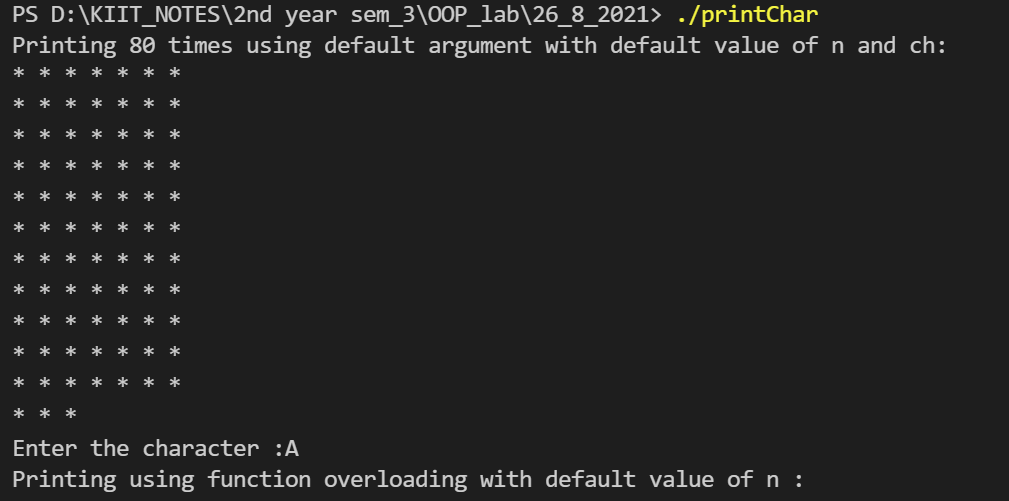
cin >> cha;

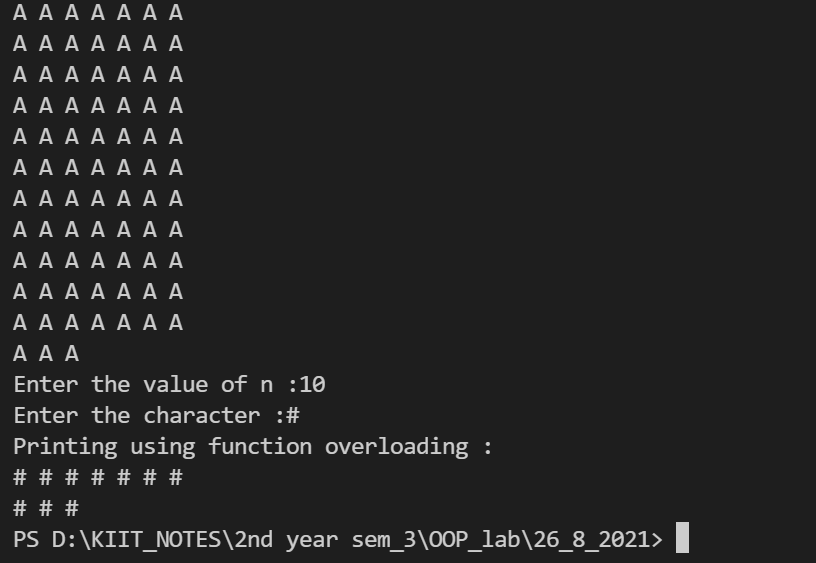
cout << "Printing using function overloading :" << endl;

obj.display(cha, n);

return 0;

}





***Question 3 :WAP to find square and cube of a number using inline function.***

#include <iostream>

using namespace std;

class cal

{

public:

inline int square(int n);

inline int cube(int n);

};

int cal::square(int n)

{

return n \* n;

}

int cal::cube(int n)

{

return n \* n \* n;

}

int main()

{

int n;

cout << "Enter the value of n :" << endl;

cin >> n;

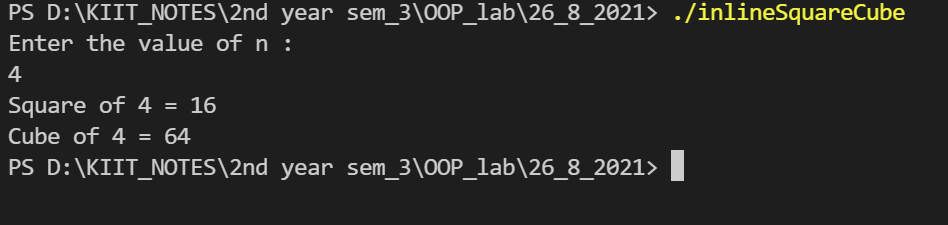
cal obj;

cout << "Square of " << n << " = " << obj.square(n) << endl;

cout << "Cube of " << n << " = " << obj.cube(n) << endl;

return 0;

}



***Question 4 :WAP to swap two integers using pass by reference.***

#include <iostream>

using namespace std;

void swap(int \*num1, int \*num2);

int main()

{

int num1, num2;

cout << "Enter 1st integer :" << endl;

cin >> num1;

cout << "Enter 2nd integer :" << endl;

cin >> num2;

swap(&num1, &num2);

cout << "num1 = " << num1 << endl;

cout << "num2 = " << num2 << endl;

return 0;

}

void swap(int \*num1, int \*num2)

{

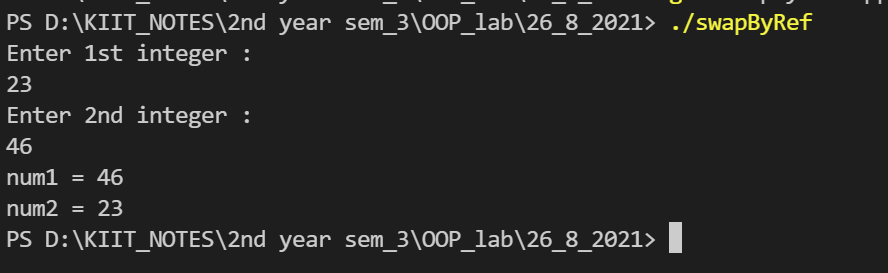
int temp;

temp = \*num1;

\*num1 = \*num2;

\*num2 = temp;

}



***Question 5 :WAP to increment the value of an argument given to a function.***

#include <iostream>

using namespace std;

class value

{

public:

int fun(int n);

};

int value::fun(int n){

return ++n;

}

int main()

{

value obj;

int n;

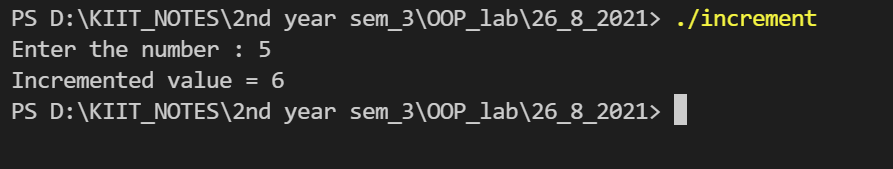
cout << "Enter the number : ";

cin >> n;

cout << "Incremented value = " << obj.fun(n) << endl;

return 0;

}



***Question 6 : Create a class distance which stores a distance in feet and inches. Input 2 distance values in objects , add them,store the resultant distance in the object and display it.***

***[write the above program in two ways.***

1. ***store the resultant distance in the calling object : c3.add(c1,c2)***
2. ***Return resultant object c3 = c1.add(c2)***

#include <iostream>

using namespace std;

class Distance

{

int dFeet;

int dInch;

public:

void input()

{

cout << "enter distance in feet :" << endl;

cin >> dFeet;

cout << "enter distance in inch :" << endl;

cin >> dInch;

}

Distance calculate(Distance c2)

{

Distance obj;

obj.dFeet = dFeet + c2.dFeet;

obj.dInch = dInch + c2.dInch;

if (dInch >= 12)

{

obj.dInch = obj.dInch - 12;

obj.dFeet++;

}

return obj;

}

void add(Distance c1, Distance c2)

{

dFeet = c1.dFeet + c2.dFeet;

dInch = c1.dInch + c2.dInch;

if (dInch >= 12)

{

dInch = dInch - 12;

dFeet++;

}

cout << "using c3.add(c1,c2) :" << endl;

cout << "distance in feet :" << dFeet << endl;

cout << "enter distance in inch :" << dInch << endl;

}

void display()

{

cout << "using c3= c1.calculate(c2) :" << endl;

cout << "distance in feet :" << dFeet << endl;

cout << "enter distance in inch :" << dInch << endl;

}

};

int main()

{

Distance c1, c2, c3;

c1.input();

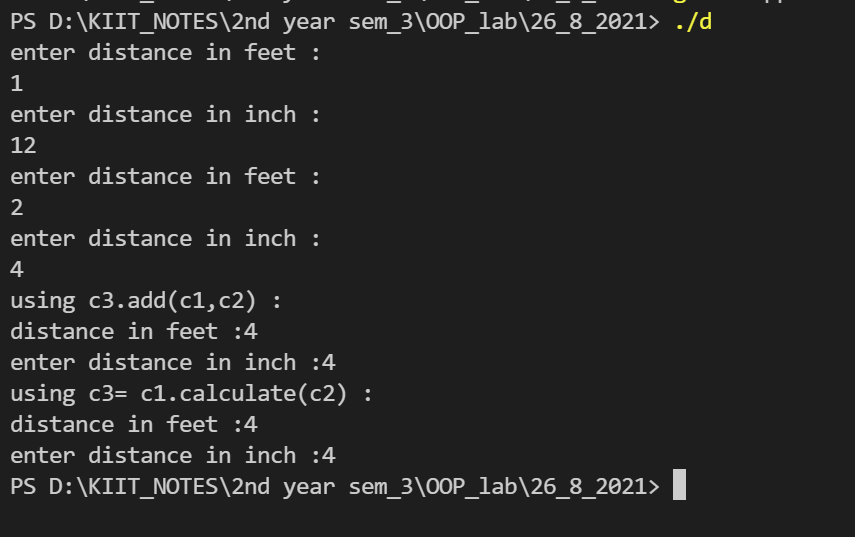
c2.input();

c3 = c1.calculate(c2);

c3.add(c1, c2);

c3.display();

}



***Question 7 : WAP to create a class LIFE with data member : manufacturing year , expire year , year to calculate life of the product.Take an input function to initialize the data members manufacturing year and expire year using a function calculate life of the product through***

***1) pass by value***

***2) pass by address***

***3) pass by reference***

#include <iostream>

using namespace std;

class life

{

public:

void inputVal(int mYear, int eYear);

void inputAdd(int \*mYear, int \*eYear);

void inputRef(int &manuYear, int &expYear);

void Valecalculate();

private:

int manufacturingYear;

int expiryYear;

int year;

};

void life::inputVal(int mYear, int eYear)

{

manufacturingYear = mYear;

expiryYear = eYear;

}

void life::inputAdd(int \*mYear, int \*eYear)

{

manufacturingYear = \*mYear;

expiryYear = \*eYear;

}

void life::inputRef(int &manuYear, int &expYear)

{

manufacturingYear = manuYear;

expiryYear = expYear;

}

void life::Valecalculate()

{

year = expiryYear - manufacturingYear ;

cout << "Life of machine = " << year << endl;

}

int main()

{

life obj;

int mYear, eYear;

cout << "Enter year of manufacturing :";

cin >> mYear;

cout << "Enter year of expiry :";

cin >> eYear;

obj.inputVal(mYear, eYear);

cout << "Call by value : " << endl;

obj.Valecalculate();

cout << "Call by Address : " << endl;

obj.inputAdd(&mYear, &eYear);

obj.Valecalculate();

cout << "Call by Reference : " << endl;

obj.inputVal(mYear, eYear);

obj.Valecalculate();

return 0;

}

