**1.Consider we have a Cars under which Santro Xing, Alto and WagonR represents individual objects. In this context each Car Object will have its price and Year of Manufacture. Write a code to implement the concepts of l-value and r-value that will display the price and manufacturer of car.**

#include<iostream>

using namespace std;

class Santro{

double sprice=130000;

int syear=2013;

public:

void displays()

{

cout<<"\n Santro price is"<<sprice;

cout<<"\n Santro manufacturing year is"<<syear;

}

};

class alto{

double aprice=120000;

int ayear=2012;

public:

void displaya()

{

cout<<"\n Alto price is"<<aprice;

cout<<"\n Altoo manufacturing year is"<<ayear;

}

};

class wagonar{

double wprice=150000;

int wyear=2017;

public:

void displayw()

{

cout<<"\n Alto price is"<<wprice;

cout<<"\n Altoo manufacturing year is"<<wyear;

}

};

int main()

{

Santro s;

alto a;

wagonar w;

s.displays();

a.displaya();

w.displayw();

}

**2.A class of 'n' students take an annual examination in 'm' subjects. Implement a program to read the marks obtained by each student in various subjects and to compute and print the total marks obtained by each of them. To implement this use concepts of l-value and r-value.**

#include<iostream>

using namespace std;

class marksheet{

int no\_of\_subject[4];

int total=0;

int i,j;

public:

void getdata()

{

for(i=0;i<=5;i++)

{

cout<<"\n enter marks of student";

for(j=0;j<=5;j++)

{

cin>>no\_of\_subject[i];

total=total+no\_of\_subject[i];

}

cout<<"\n total is"<<total;

}

}

};

int main()

{

marksheet m;

m.getdata();

}

**or**

#include<iostream>  
using namespace std;  
int main()  
{  
    int no\_of\_students=5,sum=0,i;  
    int no\_of\_subjects=5;  
    int Maths,DSC,CAO,PIPL,JAVA;  
    cout<<"Enter the marks for each student "<<endl;  
    for(i=0;i<=5;i++)  
    {  
        cout<<"enter the marks for student "<<i+1<<endl;  
        cin>>Maths>>DSC>>CAO>>PIPL>>JAVA;  
        sum=Maths+DSC+CAO+PIPL+JAVA;  
        cout<<"total marks obtained by the student "<<i+1<<"is"<<sum<<endl;  
    }  
      
}

**5.In many applications there is a need for arrays whose sizes are set by function parameters. Unfortunately, a declaration such as this one is illegal because array bounds must be a constant expression: If you need to use a parameter to specify the size of an array, you can use new to allocate the array at the start of a function and use delete to free it at the end of the function: The problem with this approach is that it is not completely reliable when it comes to preventing memory leaks. The goal here is to create a dynamic array that will handle the situation. for this situation what will be the best solution to create dynamic arrays? Enumerate your answer with creation of dynamic memory using programming principles.**

**4.Is it possible to convert an existing data type declared in high level language into a new data type? Give your views and corresponding justifications with related illustrations of code.**

#include<iostream>

using namespace std;

int main()

{

float x=5.5;

int y=5;

y=(int)x+y;

cout<<"\n y="<<y;

int a=5;

double b=50.5;

b=(int)b+a;

cout<<"\n b"<<b;

}

**3.The Automatic or Widening conversion takes place when two data types are automatically converted.**

**This will be happens when:**

* **The two data types are compatible.**
* **When we assign value of a smaller data type to a bigger data type.**

**Implement code for the following case where data is automatically converted from one type to another**

**E.g.,**

**int-->Long-->Float-->Double**

#include<iostream>

using namespace std;

int main()

{

int x=5;

long y=6;

y=y+x;

cout<<"\n y is"<<y;

float a=3.5;

a=a+y;

cout<<"\n a is"<<a;

double b=343555;

b=a+b;

cout<<"\n b is="<<b;

}