COMPUTER SCIENCE PROJECT

CARBON CREDIT MANAGEMENT SYSTEM



M S LALITHA RAMYA

XII ‘A’

CMR NATIONAL PUBLIC SCHOOL

BOARD ROLL NO: 4613942

CERTIFICATE

*This is to certify that M S LALITHA RAMYA has satisfactorily completed the project in Computer Science titled ‘*Carbon Credit Management System’ *as prescribed by the Central Board Of Secondary Education (CBSE) as part of the laboratory requirement for All India Senior Secondary Certificate Examination (AISSCE) Course during the academic year 2013-2014.*

Signature of Internal Examiner Signature of External Examiner

Date: Date:

**M S LALITHA RAMYA**

Board Roll No: 4613942

                                                                                                    School Stamp

**INDEX**

PAGE NO

Acknowledgements 4

Introduction 5

What are Carbon Credits? 6

Measuring Carbon Credits 9

Project Description 10

Classes 12

Functions 13

Header Files 13

Flow Chart 14

Screen Shots 15

Conclusion 20

Source Code in C++ 21

Glossary 64

Bibliography 65

**Acknowledgements**

First and foremost, I would like to thank my Computer Teacher, Mrs. Aruna M for giving me the opportunity to do this project and her constant guidance and support; enthusiastic interest and encouragement helped me complete this project successfully.

Lastly, my project partners Akshayaa BR, Adithya NA and Rajeev Alekkat, without whom this project would not have been as fun and enjoyable as it had been.

I enjoyed doing the project and it has certainly strengthened my understanding of various concepts learnt in class.

Thank you

M S LALITHA RAMYA

**INTRODUCTION:**

The world is getting polluted heavily and each one of us is responsible for adding the pollution into the environment knowingly or unknowingly. It is high time we realize our unnecessary contribution to carbonizing the mother earth and take remedial steps to build back the green earth.

Many of the developed countries made it mandatory that individuals, institutions and organizations compute their positive and negative contributions to the environment and the negative contributions in excess will be penalized where as excessive positive contributions can be transferred or sold to others and make some wealth.

These positive and negative contributions to the environment from each individual or household or organization are referred to as “Carbon Credits”.

**Aim:**

To develop a C++ based computer application ‘Carbon Credit Management System’ (CCMS) for managing carbon credits of individuals, institutions and organizations.

**Objective:**

To demonstrate the learning of Object Oriented programming, C++ language, with File Handling programming for fulfilling the CBSE Class XII studies.

**Tools Used:**

Dev C++

**WHAT ARE CARBON CREDITS?**

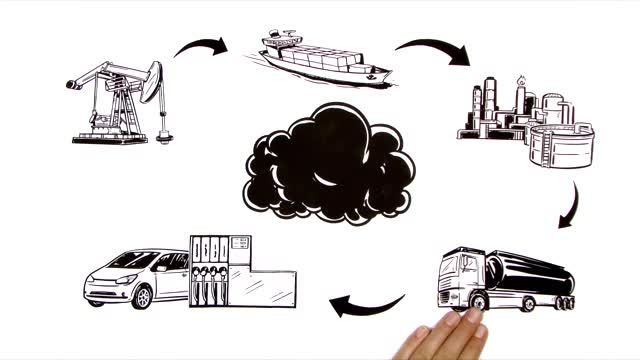
Every time we burn fossil fuels such as gas, coal or oil, carbon dioxide is released into the atmosphere. In a natural carbon cycle, carbon dioxide is re-absorbed by plants and trees. However, we are burning fuels where the carbon dioxide has been trapped under the earth's surface for millions of years, and we're doing it so quickly that plants and trees that are alive now have no chance of soaking it up and further we're cutting down rainforests.

From keeping warm in our house, to fuelling our cars, to growing our food, to manufacturing our MP3 players, energy is used. It is either burned directly (gas is burnt in your boiler for example, and petrol is burnt in your car) or it is burnt in a power station to drive turbines, which generate electricity. Fossil fuels are also burnt at various stages in the process of creating food, products and services for our consumption. The total carbon, which we as individuals are responsible for is called our carbon footprint.



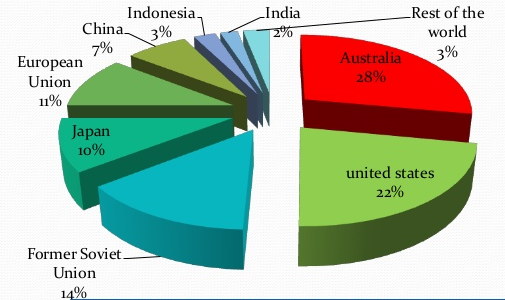
The effect of all this extra carbon dioxide in the atmosphere is that the overall temperature of the planet is increasing which is called ‘global warming’. Whilst the average global temperature is increasing, on a day-to-day level the climate is changing in unpredictable ways (from floods and hurricanes to heat waves and droughts). To try and reduce the risk of ever more extreme weather, we need to reduce how much fossil fuel we are burning.



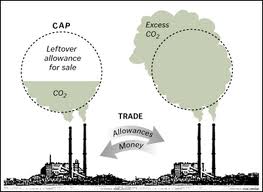


A **carbon credit** is a generic term for representing the right to emit one ton of carbon dioxide or the mass of another green house equivalent to one ton of carbon dioxide.

The concept of carbon credits came into existence as a result of increasing awareness of the need for controlling emissions.

****

Based on the Carbon credits generated, individuals and organizations may be contributing excess Caron dioxide than the average permeable limits, in which case it is said that the carbon credits possessed by the individual or organization goes negative.

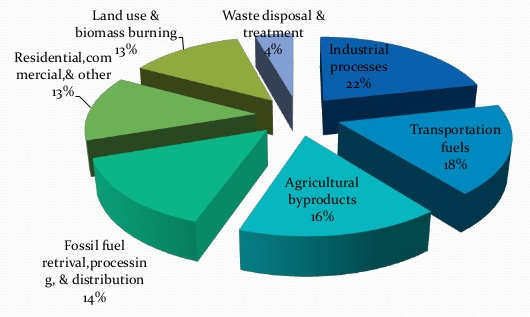


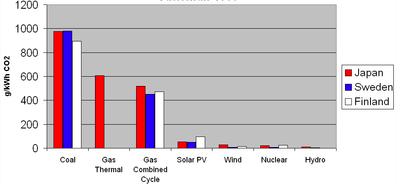
The Kyoto Protocol is an international treaty that sets binding obligations on industrialized countries to reduce emissions of greenhouse gases.

The goal is to allow market mechanisms to drive industrial and commercial processes in the direction of low emissions or less carbon intensive approaches

**MEASURING CARBON CREDITS:**

There are 3rd party agencies measuring the carbon credits based on the fuel consumption and contribution back to the environment through forestation, sing solar and wind energies etc.

****

****

For simplicity, the project created at CMR National Public School uses some standard conversion factors derived through discussion from the project guide.

**PROJECT DESCRIPTION:**

The objective is to develop a program for collecting the energy consumption and green area of certain categories and calculating the Carbon Credits for those categories.

This project is developed in C++ and the program is divided into 4 sections:

1) HOUSEHOLD.

2) COMPANIES.

3) UNIVERSITIES.

4) FACTORIES.

The program works with the help of 4 binary files that handle each of these sections.

It has separate classes for each section and also includes classes to calculate energy consumed by light and heavy-duty machines.

**Functioning:**

The program basically accepts the specific and detailed information of each of its clients, which is systematically stored in the file for future use. The information stored is then used to calculate the credits that the client earns or loses.

The carbon credits are calculated on a monthly basis; hence the details of the client can be modified every month. Provision for this has been done. The client can view the details of his or her University or Company etc. whenever they want too. Their detailed information can be modified.

The Carbon Credits can be both positive or negative depending on the emissions made by the client in the previous month. These credits can be converted into money; hence this acts as a motivation for the people to have a proper check on their emissions.

CCMS program allows the users to traverse from one category to the other.

Each client has a unique code that makes it easier for the used to search for specific data and modify them.

The program is protected by a password that only the office members are aware of. Hence illegal access is prohibited.

This program thus keeps a monthly check on the clients and helps them reduce the pollution they cause to this environment and as a result we can make our world a better and cleaner place to live in.

**CLASSES:**

|  |  |
| --- | --- |
| **NAME** | **DESCRIPTION** |
| Company() | Contains input and output functions, which help adding details into the company file and displaying form it. |
| Household() | Contains input and output functions, which help adding details into the household file and displaying form it. |
| Universities() | Contains input and output functions, which help adding details into the university file and displaying form it. |
| Factory() | Contains input and output function, which help adding details into the factory file and displaying form it. |
| LightMachinery() | Base class inherited by all the above classes. Includes functions to accept details of the light machinery and calculate credits. |
| HeavyMachinery() | Base class inherited by Factory() classes. Includes functions to accept details of the light machinery and calculate credits. |

**FUNCTIONS:**

|  |  |
| --- | --- |
| **NAME** | **DESCRIPTION** |
| float calculatearea(float,float) | used to calculate ratio of green area to total area and return credits. |
| float usage(float,float,float,float) | used to calculate the total usage and return the calculated credits pertaining to factory. |
| float usage(float,float,float) | used to calculate the total usage and return the calculated credits pertaining to company, university and household. |

**HEADER FILES:**

* iostream
* fstream
* conio.h
* string.h
* stdio.h
* stdlib.h

**FLOW CHART**

PASSWORD CHECK

MAIN MENU

HOUSEHOLD

COMPANY

UNIVERSITY

FACTORY

SUB MENU

ADD

SEARCH

DISPLAY

MODIFY

Password Incorrect (3 attempts)

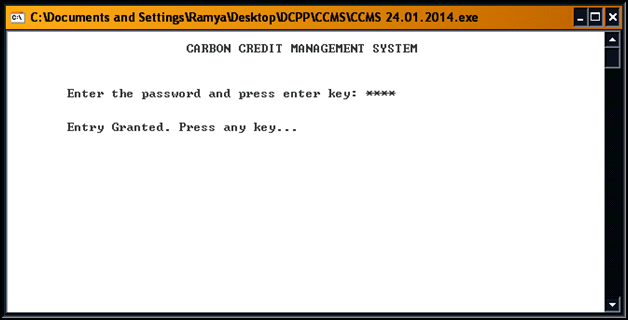
Password Correct

Goto Main Menu

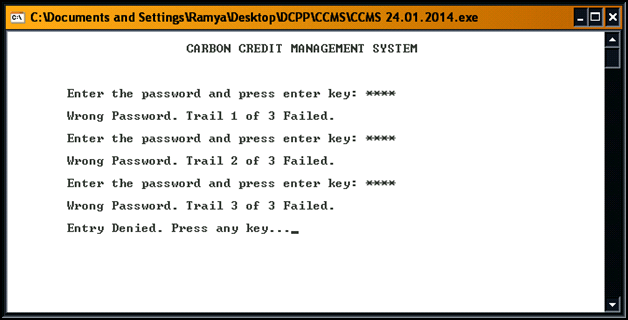
**SCREEN SHOTS**

The Screen shots for Login are…

Right Password

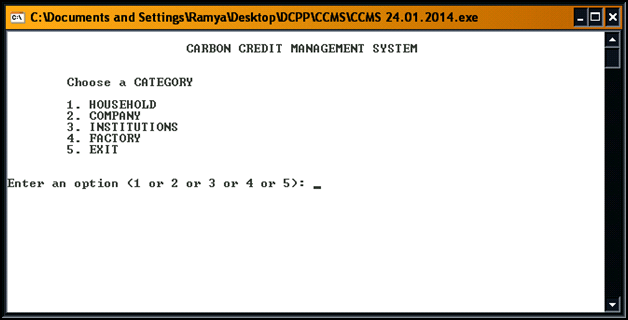


Wrong Password

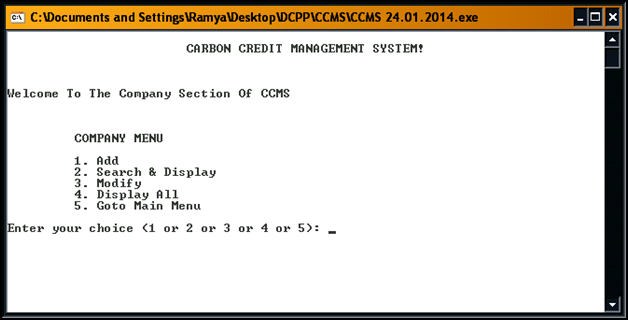


Screen shots for Menus

Main Menu

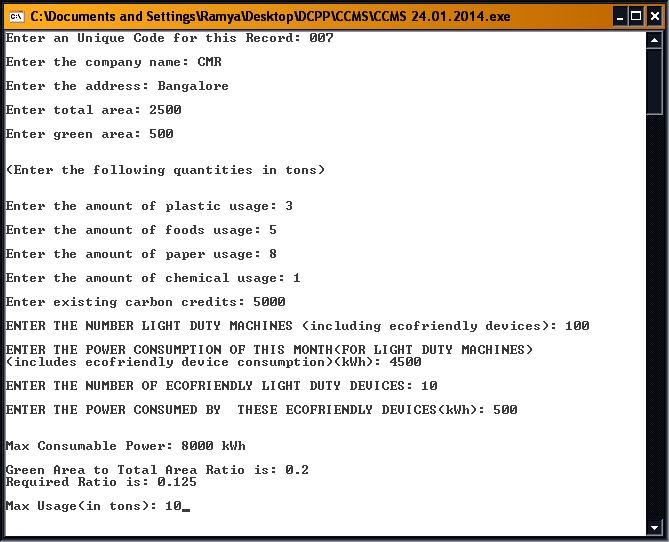


Sub Menu for Companies

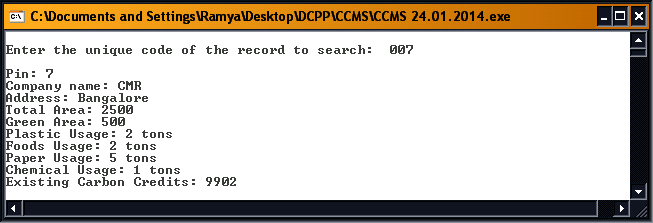


Add and Search Screen Shots for Companies

Add

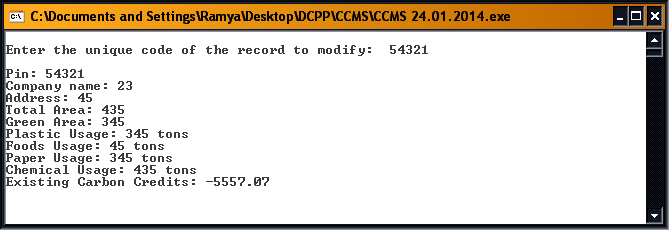


Search and Display

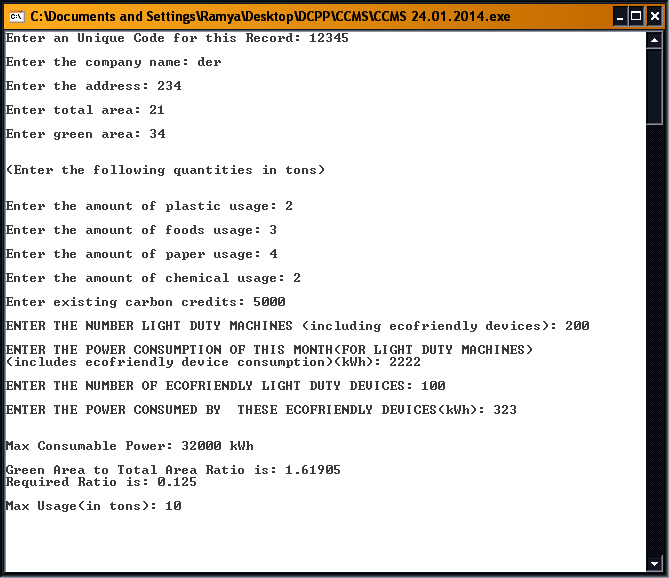


Modify Companies Screen Shots

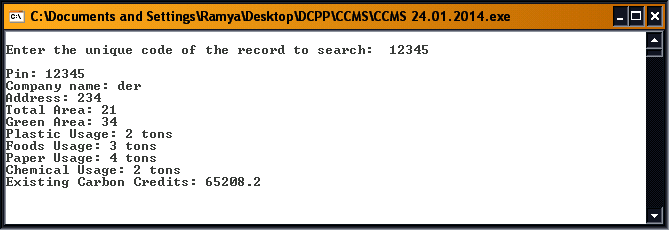
Modify 1



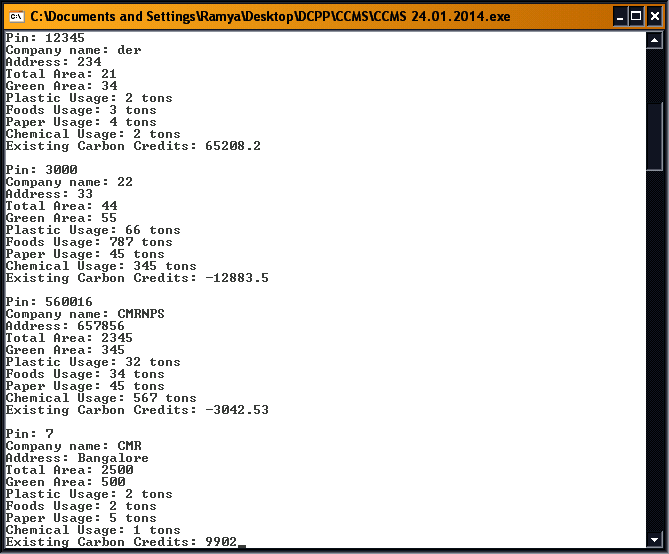
Modify 2



Modify 3



Display All Screen Shot



**CONCLUSION:**

This project helps keep a systematic check on the factories, companies, households and universities that cause harm to the environment. Carbon Crediting is a wonderful system that will help the clients to have more focus towards the damage that their establishment does to the environment.

They will work towards reducing this damage and as a reward they earn carbon credits.

According to us, The **CARBON CREDITING MANAGEMENT SYSTEM** is the best way to reduce the pollution and harm done to our environment, and we are positive that such a system would change, or rather improve, the current situation of our planet.

**SOURCE CODE in C++:**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Project: Carbon Credit Management System

Project Developed By:

MS Lalitha Ramya

Akshayaa BR

Rajeev AV

Adithya NA

Class XII, CMR National Public School (2013-14 batch)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

// include header files

#include<iostream>

#include<fstream>

#include<conio.h>

#include<string.h>

#include<stdio.h>

#include<stdlib.h>

using namespace std;

//constant variable initialization

float rr=0.125;

float max\_tons=10.0;

float m\_tons=1.0;

//Function declaration

float calculatearea(float greenarea,float totalarea);

float usage(float usage\_plastic,float usage\_food, float usage\_paper);

float usage(float usage\_plastic,float usage\_food,float usage\_paper,float usage\_chemicals);

class HeavyMachinery

{

public:

float nofm;

float pwcon;

float mconspw;

float noem;

float pwcone;

float credits;

float extra;

float calcmaxpw()

{

if(nofm<=20)

{

mconspw=nofm\*100;

}

else if(nofm<=100)

{

mconspw=nofm\*800;

}

else if(nofm<=200)

{

mconspw=nofm\*1600;

}

else

{

mconspw=nofm\*2000;

}

if(pwcon>mconspw)

{

extra=pwcon-mconspw;

credits=(-1)\*extra\*2;

}

else

{

extra=mconspw-pwcon;

credits=extra\*2;

}

cout<<"\n\nMax Consumable Power: "<<mconspw<<" kWh";

return credits;

}

void accept()

{

cout<<"\nENTER THE NUMBER OF MACHINES (including ecofriendly devices): ";

cin>>nofm;

cout<<"\nENTER THE POWER CONSUMPTION (includes ecofriendly device consumption)(kWh): ";

cin>>pwcon;

cout<<"\nENTER THE NUMBER OF ECOFRIENDLY DEVICES: ";

cin>>noem;

cout<<"\nENTER THE POWER CONSUMED BY ECOFRIENDLY DEVICES(kWh): ";

cin>>pwcone;

pwcon=pwcon-pwcone;

}

};

class LightMachinery

{

public:

float nofm;

float pwcon;

float mconspw;

float noem;

float pwcone;

float credits;

float extra;

float calcmaxpw()

{

if(nofm<=20)

{

mconspw=nofm\*20;

}

else if(nofm<=100)

{

mconspw=nofm\*80;

}

else if(nofm<=200)

{

mconspw=nofm\*160;

}

else

{

mconspw=nofm\*200;

}

if(pwcon>mconspw)

{

extra=pwcon-mconspw;

credits=(-1)\*extra\*2;

}

else

{

extra=mconspw-pwcon;

credits=extra\*2;

}

cout<<"\n\nMax Consumable Power: "<<mconspw<<" kWh";

return credits;

}

void accept()

{

cout<<"\nENTER THE NUMBER LIGHT DUTY MACHINES (including ecofriendly devices): ";

cin>>nofm;

cout<<"\nENTER THE POWER CONSUMPTION OF THIS MONTH(FOR LIGHT DUTY MACHINES)\n"<<"(includes ecofriendly device consumption)(kWh): ";

cin>>pwcon;

cout<<"\nENTER THE NUMBER OF ECOFRIENDLY LIGHT DUTY DEVICES: ";

cin>>noem;

cout<<"\nENTER THE POWER CONSUMED BY THESE ECOFRIENDLY DEVICES(kWh): ";

cin>>pwcone;

pwcon=pwcon-pwcone;

}

};

class company:public LightMachinery

{

public://General Attributes

char company\_name[30];

char address[100]; // Different credits for URBAN, RURAL and FOREST areas

float tcredits,total\_area, green\_area; // in square feet

int no\_of\_employees;

float credits;

int pin;

//Negative contribution Attributes

// utility consumption in units

float usage\_plastic, usage\_food, usage\_paper, usage\_chemicals; // usage quantity in Tons

void get\_com\_data()

{

cout<<"Enter an Unique Code for this Record: ";

cin>>pin;

cout<<"\nEnter the company name: ";

cin>>company\_name;

cout<<"\nEnter the address: ";

cin>>address;

cout<<"\nEnter total area: ";

cin>>total\_area;

cout<<"\nEnter green area: ";

cin>>green\_area;

cout<<"\n\n(Enter the following quantities in tons)\n\n";

cout<<"\nEnter the amount of plastic usage: ";

cin>>usage\_plastic;

cout<<"\nEnter the amount of foods usage: ";

cin>>usage\_food;

cout<<"\nEnter the amount of paper usage: ";

cin>>usage\_paper;

cout<<"\nEnter the amount of chemical usage: ";

cin>>usage\_chemicals;

cout<<"\nEnter existing carbon credits: ";

cin>>tcredits;

LightMachinery::accept();

credits=LightMachinery::calcmaxpw();

credits= credits+calculatearea(green\_area,total\_area);

credits= credits+usage(usage\_plastic,usage\_food,usage\_paper,usage\_chemicals);

tcredits=tcredits+credits;

}

void put\_com\_data()

{

cout<<"\nPin: ";

cout<<pin;

cout<<"\nCompany name: ";

cout<<company\_name;

cout<<"\nAddress: ";

cout<<address;

cout<<"\nTotal Area: ";

cout<<total\_area;

cout<<"\nGreen Area: ";

cout<<green\_area;

cout<<"\nPlastic Usage: ";

cout<<usage\_plastic<<" tons";

cout<<"\nFoods Usage: ";

cout<<usage\_food<<" tons";

cout<<"\nPaper Usage: ";

cout<<usage\_paper<<" tons";

cout<<"\nChemical Usage: ";

cout<<usage\_chemicals<<" tons";

cout<<"\nExisting Carbon Credits: ";

cout<<tcredits;

}

int getpin()

{ return pin;}

};

class household:public LightMachinery

{

public:

int nf\_ppl;

int pin;

char address[100]; //includes petrol,diesel,lpg,etc..

float credits,tcredits;

float usage\_plastic, usage\_food, usage\_paper;

float green\_area,total\_area;

void details()

{

cout<<"ENTER PIN CODE: ";

cin>>pin;

cout<<"\nEnter address: ";

cin>>address;

cout<<"\nEnter the no. of ppl in the house: ";

cin>>nf\_ppl;

cout<<"\nEnter total living space(total area): ";

cin>>total\_area;

cout<<"\nEnter the area of green space in and around the house: ";

cin>>green\_area;

cout<<"\n\n(Enter the following quantities in tons)\n\n";

cout<<"\nEnter the amount of plastic usage: ";

cin>>usage\_plastic;

cout<<"\nEnter the amount of foods usage: ";

cin>>usage\_food;

cout<<"\nEnter the amount of paper usage: ";

cin>>usage\_paper;

cout<<"\nEnter existing carbon credits: ";

cin>>tcredits;

LightMachinery::accept();

credits=LightMachinery::calcmaxpw();

credits= credits+calculatearea(green\_area,total\_area);

credits= credits+usage(usage\_plastic,usage\_food,usage\_paper);

tcredits=tcredits+credits;

}

void display()

{

cout<<"\nPin: ";

cout<<pin;

cout<<"\nAddress: ";

cout<<address;

cout<<"\nTotal Area: ";

cout<<total\_area;

cout<<"\nGreen Area: ";

cout<<green\_area;

cout<<"\nPlastic Usage: ";

cout<<usage\_plastic<<" tons";

cout<<"\nFoods Usage: ";

cout<<usage\_food<<" tons";

cout<<"\nPaper Usage: ";

cout<<usage\_paper<<" tons";

cout<<"\nExisting Carbon Credits: ";

cout<<tcredits;

}

int getpin()

{ return pin;}

}H;

class universities: public LightMachinery

{

public:

char name\_uni[20], address\_uni[100];

int working\_hrs;

int pin;

int no\_of\_classes ;

float green\_area,total\_area;

float credits,tcredits;

float usage\_plastic, usage\_food, usage\_paper, usage\_chemicals;

void getdata()

{

cout<<"ENTER PIN CODE: ";

cin>>pin;

cout<<"\nEnter the name of the university: ";

cin>>name\_uni;

cout<<"\nAddress of the university: ";

cin>>address\_uni;

cout<<"\nEnter daily working hours: ";

cin>>working\_hrs;

cout<<"\nEnter number of classes: ";

cin>>no\_of\_classes;

cout<<"\nEnter total area: ";

cin>>total\_area;

cout<<"\nEnter green area: ";

cin>>green\_area;

cout<<"\n\n(Enter the following quantities in tons)\n\n";

cout<<"\nEnter the amount of plastic usage: ";

cin>>usage\_plastic;

cout<<"\nEnter the amount of foods usage: ";

cin>>usage\_food;

cout<<"\nEnter the amount of paper usage: ";

cin>>usage\_paper;

cout<<"\nEnter the amount of chemical usage: ";

cin>>usage\_chemicals;

cout<<"\nEnter existing carbon credits: ";

cin>>tcredits;

LightMachinery::accept();

credits=LightMachinery::calcmaxpw();

credits= credits+calculatearea(green\_area,total\_area);

credits= credits+usage(usage\_plastic,usage\_food,usage\_paper,usage\_chemicals);

tcredits=tcredits+credits;

}

void showdata()

{

cout<<"\nPin: ";

cout<<pin;

cout<<"\nUniversity name: ";

cout<<name\_uni;

cout<<"\nAddress: ";

cout<<address\_uni;

cout<<"\nDaily working hours: ";

cout<<working\_hrs;

cout<<"\nNumber of classes: ";

cout<<no\_of\_classes;

cout<<"\nTotal Area: ";

cout<<total\_area;

cout<<"\nGreen Area: ";

cout<<green\_area;

cout<<"\nPlastic Usage: ";

cout<<usage\_plastic<<" tons";

cout<<"\nFoods Usage: ";

cout<<usage\_food<<" tons";

cout<<"\nPaper Usage: ";

cout<<usage\_paper<<" tons";

cout<<"\nChemical Usage: ";

cout<<usage\_chemicals<<" tons";

cout<<"\nExisting Carbon Credits: ";

cout<<tcredits;

}

int getpin()

{ return pin;}

}U;

class Factory: public HeavyMachinery,public LightMachinery

{

public:

char name[30];

int pin;

char addrs[100];

float credits;

float tcredits,tarea,garea;

float usage\_plastic, usage\_food, usage\_paper, usage\_chemicals;

float noemp;

void input()

{

cout<<"ENTER PIN CODE: ";

cin>>pin;

cout<<"\nENTER NAME OF FACTORY: ";

cin>>name;

cout<<"\nENTER ADDRESS: ";

cin>>addrs;

cout<<"\nENTER TOTAL AREA: ";

cin>>tarea;

cout<<"\nENTER GREEN AREA: ";

cin>>garea;

cout<<"\nENTER NUMBER OF EMPLOYEES: ";

cin>>noemp;

cout<<"\n\n(Enter the following quantities in tons)\n\n";

cout<<"\nEnter the amount of plastic usage: ";

cin>>usage\_plastic;

cout<<"\nEnter the amount of foods usage: ";

cin>>usage\_food;

cout<<"\nEnter the amount of paper usage: ";

cin>>usage\_paper;

cout<<"\nEnter the amount of chemical usage: ";

cin>>usage\_chemicals;

cout<<"\nEnter existing carbon credits: ";

cin>>tcredits;

LightMachinery::accept();

HeavyMachinery::accept();

credits=LightMachinery::calcmaxpw();

credits= credits+HeavyMachinery::calcmaxpw();

credits= credits+calculatearea(garea,tarea);

credits= credits+usage(usage\_plastic,usage\_food,usage\_paper,usage\_chemicals);

tcredits=tcredits+credits;

}

void display()

{

cout<<"\nPin: ";

cout<<pin;

cout<<"\nFactory name: ";

cout<<name;

cout<<"\nAddress: ";

cout<<addrs;

cout<<"\nTotal Area: ";

cout<<tarea;

cout<<"\nGreen Area: ";

cout<<garea;

cout<<"\nPlastic Usage: ";

cout<<usage\_plastic<<" tons";

cout<<"\nFoods Usage: ";

cout<<usage\_food<<" tons";

cout<<"\nPaper Usage: ";

cout<<usage\_paper<<" tons";

cout<<"\nChemical Usage: ";

cout<<usage\_chemicals<<" tons";

cout<<"\nExisting Carbon Credits: ";

cout<<tcredits;

}

int getpin()

{ return pin;}

}F;

// main() function

int main()

{

int ch, flag, pwcount=0,exitflag=0;

int ctrh=0, pos=0, hflag;

//variables for PASSWORD

char pass[4+1],ch1;

int j = 0;

system("cls");

cout<<"\n\t\t\tCARBON CREDIT MANAGEMENT SYSTEM\n\n";

do

{

cout<<"\n\n\tEnter the password and press enter key: ";

// 4 character password

for(j=0;j<4;j++)

{

pass[j] = getch();

cout<<"\*";

}

pass[4]='\0';

//cout<<pass;

getch();

if(strcmpi(pass,"rara")==0)

{

pwcount=3;

}

else

{

cout<<"\n\n\tWrong Password. Trail "<<pwcount+1<<" of 3 Failed.";

pwcount++;

exitflag++;

}

}while(pwcount!=3);

if(exitflag < 3)

{

cout<<"\n\n\n\tEntry Granted. Press any key...";

getch();

}

else

{

cout<<"\n\n\tEntry Denied. Press any key...";

getch();

return 0; // exiting if the password trails fail for 3 times

}

// Selecting the category type for Carbon Credit <aManagement

do

{

system("cls");

cout<<"\n\t\t\tCARBON CREDIT MANAGEMENT SYSTEM\n\n";

cout<<"\n\tChoose a CATEGORY\n";

cout<<"\n\t1. HOUSEHOLD";

cout<<"\n\t2. COMPANY";

cout<<"\n\t3. INSTITUTIONS";

cout<<"\n\t4. FACTORY";

cout<<"\n\t5. EXIT";

cout<<"\n\n\nEnter an option (1 or 2 or 3 or 4 or 5): ";

cin>>ch;

switch(ch)

{

case 1: {

system("cls");

cout<<"\n\t\t\tCARBON CREDIT MANAGEMENT SYSTEM!";

int choice;

cout<<"\n\n\n\nWelcome To The Household Section Of CCMS\n\n";

do

{

cout<<"\n\t HOUSEHOLD MENU\n";

cout<<"\n\t 1. Add";

cout<<"\n\t 2. Search & Display";

cout<<"\n\t 3. Modify";

cout<<"\n\t 4. Display All";

cout<<"\n\t 5. Goto Main Menu";

cout<<"\n\n\t Enter your choice (1 or 2 or 3 or 4 or 5): ";

cin>>choice;

switch(choice)

{

case 1: // Add Records

{

system("cls");

ofstream hse1;

hse1.open("hdetails.dat",ios::app|ios::binary);

H.details();

hse1.write((char\*)&H,sizeof(H));

cin.get();

hse1.close();

break;

}

case 2: // Search and Display Records

{

int spin; // Pin to Search

flag=0;

system("cls");

cout<<"\nEnter the unique code of the record to search: ";

cin>>spin;

ifstream hse2;

hse2.open("hdetails.dat",ios::in|ios::binary);

hse2.seekg(0);

while(!hse2.eof())

{

hse2.read((char\*)&H,sizeof(H));

if(H.getpin()==spin) //getpin() is a func that returns the pin

{

H.display();

flag++;

break;

}

}

if(flag==0)

{

cout<<"\n\n Match not Found!! ";

}

hse2.close();

getch();

break;

}

case 3: // Modify records

{

int mpin, pos, insertpos; // mpin-pin to modify

int flag = 0;

system("cls");

cout<<"\nEnter the unique code of the record to modify: ";

cin>>mpin;

ifstream hse3;

hse3.open("hdetails.dat",ios::in|ios::binary);

hse3.seekg(0);

while(!hse3.eof())

{

pos=hse3.tellg();

hse3.read((char\*)&H,sizeof(H));

if(H.getpin()==mpin) //getpin() is a func that returns the pin

{

insertpos = pos;

H.display();

flag++;

}

}

hse3.close();

if(flag==0)

cout<<"\nRecord not found !! \n";

else

{

fstream hse3;

hse3.open("hdetails.dat",ios::in|ios::out|ios::binary);

getch();

hse3.seekg(insertpos);

cout<<"\nPress any key to modify the above record\n";

getch();

system("cls");

H.details();

hse3.seekg(insertpos);

hse3.write((char\*)&H,sizeof(H));

cin.get();

hse3.close();

}

getch();

break;

}

case 4: // Display all records

{

system("cls");

ifstream hse4;

hse4.open("hdetails.dat",ios::in|ios::binary);

hse4.seekg(0);

while(!hse4.eof())

{

hse4.read((char\*)&H,sizeof(H));

H.display();

cin.get();

}

hse4.close();

getch();

break;

}

case 5:

{break;}

default:

{cout<<"\n\nINVALID ENTRY!! TRY AGAIN!!!\n\n";

break;

}

}

}

while(choice!=5);

break;

}

case 2: {

int opt;

do

{

system("cls");

cout<<"\n\t\t\tCARBON CREDIT MANAGEMENT SYSTEM!";

cout<<"\n\n\n\nWelcome To The Company Section Of CCMS\n\n";

cout<<"\n\n\t COMPANY MENU\n";

cout<<"\n\t 1. Add";

cout<<"\n\t 2. Search & Display";

cout<<"\n\t 3. Modify";

cout<<"\n\t 4. Display All";

cout<<"\n\t 5. Goto Main Menu";

cout<<"\n\nEnter your choice (1 or 2 or 3 or 4 or 5): ";

cin>>opt;

company C; // Company Object

switch(opt)

{

case 1: // Add Records

{

system("cls");

ofstream cmp1;

cmp1.open("cdetails.dat",ios::app|ios::binary);

C.get\_com\_data();

cmp1.write((char\*)&C,sizeof(C));

cin.get();

cmp1.close();

break;

}

case 2: // Search and Display Records

{

int pin2;

flag=0;

system("cls");

cout<<"\nEnter the unique code of the record to search: ";

cin>>pin2;

ifstream cmp2;

cmp2.open("cdetails.dat",ios::in|ios::binary);

cmp2.seekg(0);

while(!cmp2.eof())

{

cmp2.read((char\*)&C,sizeof(C));

if(C.getpin()==pin2) //getpin() is a func that returns the pin

{

C.put\_com\_data();

flag++;

break;

}

}

if(flag==0)

{

cout<<"\n\n Match not Found!! ";

}

cmp2.close();

getch();

break;

}

case 3: // Modify records

{

int pin3, pos, insertpos;

int flag = 0;

system("cls");

cout<<"\nEnter the unique code of the record to modify: ";

cin>>pin3;

ifstream cmp3;

cmp3.open("cdetails.dat",ios::in|ios::binary);

cmp3.seekg(0);

while(!cmp3.eof())

{

pos=cmp3.tellg();

cmp3.read((char\*)&C,sizeof(C));

if(C.getpin()==pin3) //getpin() is a func that returns the pin

{

insertpos = pos;

C.put\_com\_data();

flag++;

}

}

cmp3.close();

if(flag==0)

cout<<"\nRecord not found !! \n";

else

{

fstream cmp3;

cmp3.open("cdetails.dat",ios::in|ios::out|ios::binary);

getch();

cmp3.seekg(insertpos);

cout<<"\nPress any key to modify the above record\n";

getch();

system("cls");

C.get\_com\_data();

cmp3.seekg(insertpos);

cmp3.write((char\*)&C,sizeof(C));

cin.get();

cmp3.close();

}

getch();

break;

}

case 4: // Display all records

{

system("cls");

ifstream cmp4;

cmp4.open("cdetails.dat",ios::in|ios::binary);

cmp4.seekg(0);

while(!cmp4.eof())

{

cmp4.read((char\*)&C,sizeof(C));

C.put\_com\_data();

cin.get();

}

cmp4.close();

getch();

break;

}

case 5:

{break;}

default:

{cout<<"\n\nINVALID ENTRY!! TRY AGAIN!!!\n\n";

break;

}

}}

while(opt!=5);

break;

}

case 3:

{

cout<<"UNIVERSITY!";

int c;

cout<<"Welcome To The UNIVERSITY Section Of CCMS";

do

{

cout<<"\n\n Menu \n";

cout<<"\n 1.Add university";

cout<<"\n 2.Search And Display university";

cout<<"\n 3.Modify";

cout<<"\n 4.Display All universities";

cout<<"\n 5.Exit";

cout<<"\n\n Enter your choice (1 or 2 or 3 or 4 or 5) : ";

cin>>c;

switch(c)

{

case 1:

{

system("cls");

ofstream f1;

f1.open("udetails.dat",ios::app|ios::binary);

U.getdata();

f1.write((char\*)&U,sizeof(U));

cin.get();

f1.close();

break;

}

case 2:

{

int pinU;

int uflag=0;system("cls");

cout<<"\nEnter the unique code alloted so as to delete : ";

cin>>pinU;

ifstream f2;

f2.open("udetails.dat",ios::binary|ios::in|ios::out);

f2.seekg(0);

while(!f2.eof())

{

f2.read((char\*)&U,sizeof(U));

if(U.getpin()==pinU) // getpin() is the return func. which returns pin

{

U.showdata();

uflag++;

f2.close();

break;

}

}

if(uflag==0)

{

cout<<"\n\n Match not Found!! ";

}

f2.close();

getch();

break;

}

case 3:

{

system("cls");

int p,pos1,intpos,flag1=0;

cout<<"\nEnter the unique code alloted so as to delete : ";

cin>>p;

ifstream f3;

f3.open("udetails.dat",ios::binary|ios::in);

f3.seekg(0);

while(!f3.eof())

{

int pos1=f3.tellg();

f3.read((char\*)&U,sizeof(U));

if(U.getpin()==p);

{

intpos=pos1;

U.showdata();

flag1++;

}

}

f3.close();

if(flag1==0)

{cout<<"\n Record not found !! \n";}

else

{

fstream f3;

f3.open("udetails.dat",ios::in|ios::out|ios::binary);

getch();

f3.seekg(intpos);

cout<<"\nPress any key to modify the above record\n";

getch();

system("cls");

U.getdata();

f3.seekg(intpos);

f3.write((char\*)&U,sizeof(U));

cin.get();

f3.close();

}

getch();

break;

}

case 4: // Display all records

{

system("cls");

ifstream f4;

f4.open("udetails.dat",ios::in|ios::binary);

f4.seekg(0);

//ff.open("udetails.dat",ios::in|ios::binary);

while(!f4.eof())

{

f4.read((char\*)&U,sizeof(U));

U.showdata();

cin.get();

}

f4.close();

getch();

break;

}

case 5:

{break;}

default:

{cout<<"\n\nINVALID ENTRY!! TRY AGAIN!!!\n\n";

break;

}

}

}

while(c!=5);

break;

}

case 4:

{

system("cls");

int f;

int fflag;

cout<<"Welcome To The Factory Section Of CCMS";

do

{

cout<<"\n\nMenu\n";

cout<<"\n1)Add";

cout<<"\n2)Search And Display";

cout<<"\n3)Modify";

cout<<"\n4)Display All";

cout<<"\n5)Exit";

cout<<"\n\nEnter your option(1 or 2 or 3 or 4 or 5) : ";

cin>>f;

switch(f)

{

case 1: // Add Records

{

system("cls");

ofstream cmp1f;

cmp1f.open("fdetails.dat",ios::app|ios::binary);

F.input();

cmp1f.write((char\*)&F,sizeof(F));

cin.get();

cmp1f.close();

break;

}

case 2: // Search and Display Records

{

int pin2;

flag=0;

system("cls");

cout<<"\nEnter the unique code of the record to search: ";

cin>>pin2;

ifstream cmp2f;

cmp2f.open("fdetails.dat",ios::in|ios::binary);

cmp2f.seekg(0);

while(!cmp2f.eof())

{

cmp2f.read((char\*)&F,sizeof(F));

if(F.getpin()==pin2) //getpin() is a func that returns the pin

{

F.display();

flag++;

break;

}

}

if(flag==0)

{

cout<<"\n\n Match not Found!! ";

}

cmp2f.close();

getch();

break;

}

case 3: // Modify records

{

int pin3, pos, insertpos;

int flag = 0;

system("cls");

cout<<"\nEnter the unique code of the record to modify: ";

cin>>pin3;

ifstream cmp3f;

cmp3f.open("fdetails.dat",ios::in|ios::binary);

cmp3f.seekg(0);

while(!cmp3f.eof())

{

pos=cmp3f.tellg();

cmp3f.read((char\*)&F,sizeof(F));

if(F.getpin()==pin3) //getpin() is a func that returns the pin

{

insertpos = pos;

F.display();

flag++;

}

}

cmp3f.close();

if(flag==0)

cout<<"\nRecord not found !! \n";

else

{

fstream cmp3f;

cmp3f.open("fdetails.dat",ios::in|ios::out|ios::binary);

getch();

cmp3f.seekg(insertpos);

cout<<"\nPress any key to modify the above record\n";

getch();

system("cls");

F.input();

cmp3f.seekg(insertpos);

cmp3f.write((char\*)&F,sizeof(F));

cin.get();

cmp3f.close();

}

getch();

break;

}

case 4: // Display all records

{

system("cls");

ifstream cmp4f;

cmp4f.open("fdetails.dat",ios::in|ios::binary);

cmp4f.seekg(0);

//fcmp.open("cdetails.dat",ios::in|ios::binary);

while(!cmp4f.eof())

{

cmp4f.read((char\*)&F,sizeof(F));

F.display();

cin.get();

}

cmp4f.close();

getch();

break;

}

case 5:

{break;}

default:

{cout<<"\n\nINVALID ENTRY!! TRY AGAIN!!!\n\n";

break;

}

}

}

while(f!=5);

break;

}

case 5: {break;}

default:{ cout<<"\n\n\nINVALID ENTRY!!";

break;}

}

}

while(ch!=5);

/\*}

else

cout<<"\n\n\tWrong Password. Entry Denied!";

cout<<"\n\n\tPress any key to quit. THANK YOU!!!";

getch();\*/

return 0;

}

//function definitions

float usage(float usage\_plastic,float usage\_food,float usage\_paper,float usage\_chemicals)

{

float tot\_usage=usage\_plastic+ usage\_food+ usage\_paper+ usage\_chemicals;

float credits=0.0;

if(tot\_usage<=max\_tons)

{

credits=(max\_tons-(tot\_usage))\*10;

}

if(tot\_usage>max\_tons)

{

credits=(tot\_usage-max\_tons)\*(-1)\*10;

}

cout<<"\n\nMax Usage(in tons): "<<max\_tons;

return credits;

}

float usage(float usage\_plastic,float usage\_food, float usage\_paper)

{

float tot\_usage=usage\_plastic+ usage\_food+ usage\_paper;

float credits=0.0;

if(tot\_usage<=m\_tons)

{

credits=(m\_tons-(tot\_usage))\*10;

}

if(tot\_usage>m\_tons)

{

credits=(tot\_usage-m\_tons)\*(-1)\*10;

}

cout<<"\n\nMax Usage (in tons): "<<m\_tons;

return credits;

}

float calculatearea(float greenarea,float totalarea)

{

float gr=greenarea/totalarea;

cout<<"\n\nGreen Area to Total Area Ratio is: "<<gr;

cout<<"\nRequired Ratio is: "<<rr;

float credits=0.0;

if(gr>rr)

{

credits= gr\*10;

}

if(gr<rr)

{

credits=gr\*(-1)\*10;

}

return credits;

}

**GLOSSARY:**

* **CBSE -**

Central Board of Secondary Education

* **CCMS -**

Carbon Credit Management System

* **CMR NPS -**

CMR National Public School

* **Dev C++ -**

C++ Compiler by Bloodshed Software

* **File Handling -**

File Editing in binary and ASCII modes

* **GUI -**

Graphical User Interface

* **OOP -**

Object Oriented Programming

**BIBLIOGRAPHY:**

* <http://en.wikipedia.org/wiki/>
* [www.google.co.in](http://www.google.co.in)
* Computer Science with C++, Text Book XII