**MENU-DRIVEN UNIT CONVERSION CALCULATOR**

**Mini Project (SDF-II)**

Submitted by:

**Abhay Arya (9920103097)**

**Chandrika Rajvanshi (9920103096)**

**Chaynika Bisht (9920103091)**

Submitted to:

**Ms. Swati Gupta**

**Ms. Shilpa Budhkar**



**Department of CSE/IT**

**Jaypee Institute of Information Technology University, Noida**

# May 2021

*Table Of Contents*

1. *Abstract*
2. *Introduction*
3. *Objective*
4. *Class Diagram*
5. *Program*
6. *Result and Discussion*
7. *Conclusions*
8. *References*

*Abstract*

While performing calculations, be it in the field of Physics, Chemistry, Mathematics or any physical research work, the students may encounter tedious errors emerging due to incorrect usage of units hence, there is a need to accomplish an ample research to discover the intricacies involved in the entire process.

To tackle this issue, there is a need of a “Menu Driven Unit Conversion Calculator” so that all the conversions between the different physical quantities required for the extensive calculations in research work can be made easily on one screen.

This “Unit Conversion Calculator” has been designed within the boundaries of the concepts taught and learnt in the Software Development Fundamentals-II classes and serves as an easy to use tool to carry out conversions between different units and measurements.

It is a small program developed extensively to exercise what we could learn during the semester study. The project shall help identify and resolve common problems.

This menu-driven unit converting calculator eliminates the use of different devices to perform different calculations and rather, provides one single window where all kinds of conversions can be performed.

*Introduction*

The project basically uses the high level object oriented programming language- C++ to serve its purpose and is inclusive of several concepts including usage of *classes and objects, constructors, destructors, file handling, functions, strings, templates* etcetera.

* This calculator provides a list of different units of measurement and enables them to choose the one in which the desired conversions have to be made among- 1. Length, 2. Temperature, 3. Weight, 4. Area, 5. Volume.
* For instance, on selecting “2”, a list of all the possible temperature based conversions are displayed to the user to choose from, like:

- Celsius-Fahrenheit

- Fahrenheit-Celsius

- Celsius-Kelvin

- Kelvin-Celsius

* When the user wishes to exit, they may press “6” which terminates the program.

*Objective*

The main aim of the project is to create a simple device which be easily operated by anyone and contains all of the following features to make its functioning even smoother:

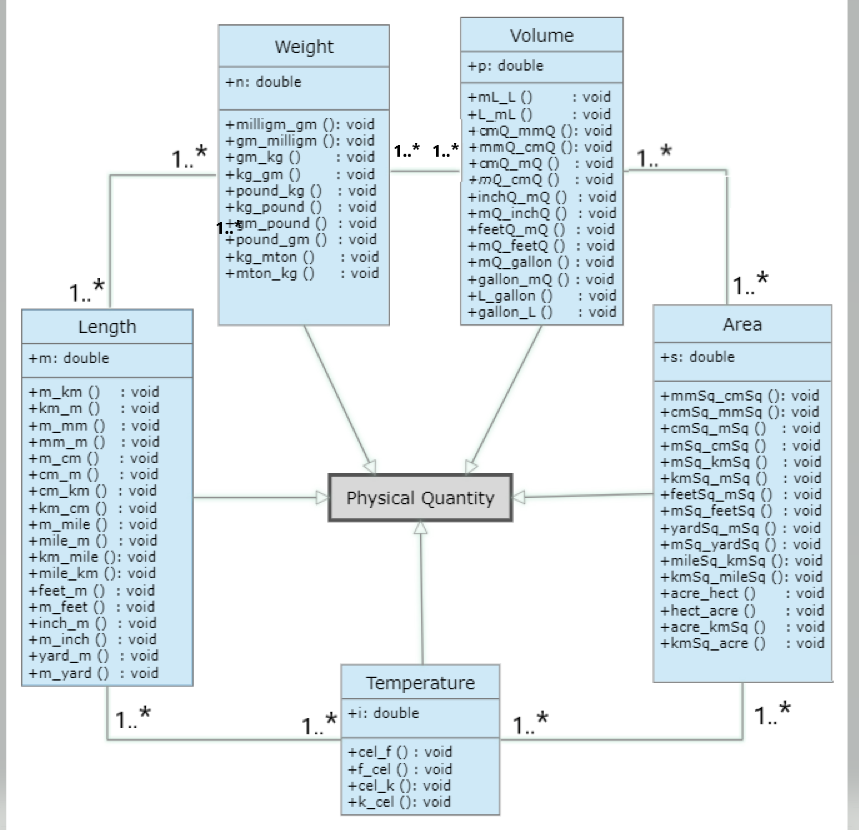
a) The programming language used i.e. C++ is simple and would make the program to run on any computer.

b) The project will ensure timely and accurate processing in computer.

c) The speed of the computer will improve after the implementation of the anticipated simple calculator.

d) Since it is menu-driven, it aims to be easy to use and understandable for the user.

*Class Diagram*



*Program*

#include<iostream> // Library

#include<fstream> //for file handling

#include<conio.h> //console input and output operations

using namespace std;

void HeadingString(string &str, int N) //Function for heading

{

for (int i = 0; i < N; i++) {

cout<< str[i]<< " ";

}

cout<<"\n \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n\n";

cout<<"This tool enables you to easily convert physical units and quantities and makes calculations very easy!\n\n";

}

class Length // Length section

{

public:

double m;

void m\_km ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Kilometer: "<<m/1000;

}

void km\_m ()

{

cout<<"Kilometer: ";

cin>>m;

cout<<"In Meter: "<<m\*1000;

}

void m\_mm ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Millimeter: "<<m\*1000;

}

void mm\_m ()

{

cout<<"Millimeter: ";

cin>>m;

cout<<"In Meter: "<<m/1000;

}

void m\_cm ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Centimeter: "<<m\*100;

}

void cm\_m ()

{

cout<<"Centimeter: ";

cin>>m;

cout<<"In Meter: "<<m/100;

}

void cm\_km ()

{

cout<<"Centimeter: ";

cin>>m;

cout<<"In Kilometer: "<<m\*0.00001;

}

void km\_cm ()

{

cout<<"Kilometer: ";

cin>>m;

cout<<"In Centimeter: "<<m/0.00001;

}

void m\_mile ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Mile: "<<m\*.00062;

}

void mile\_m ()

{

cout<<"Mile: ";

cin>>m;

cout<<"In Meter: "<<m\*1609.344;

}

void km\_mile ()

{

cout<<"Kilometer: ";

cin>>m;

cout<<"In Mile: "<<m\*.6213712;

}

void mile\_km ()

{

cout<<"Mile: ";

cin>>m;

cout<<"In kilometer: "<<m\*1.60934;

}

void feet\_m ()

{

cout<<"Feet: ";

cin>>m;

cout<<"In Meter: "<<m\*.3048;

}

void m\_feet ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Feet: "<<m\*3.28084;

}

void inch\_m ()

{

cout<<"Inch: ";

cin>>m;

cout<<"In Meter: "<<m\*.0254;

}

void m\_inch ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Inch: "<<m\*39.37008;

}

void yard\_m ()

{

cout<<"Yard: ";

cin>>m;

cout<<"In Meter: "<<m\*0.9144;

}

void m\_yard ()

{

cout<<"Meter: ";

cin>>m;

cout<<"In Yard: "<<m\*1.09361;

}

};

class Weight // Weight section

{

public:

double n;

void milligm\_gm ()

{

cout<<"Milligram: ";

cin>>n;

cout<<"In Gram: "<<n\*0.001;

}

void gm\_milligm ()

{

cout<<"Gram: ";

cin>>n;

cout<<"In Milligram: "<<n\*1000;

}

void pound\_kg ()

{

cout<<"Pound: ";

cin>>n;

cout<<"In Killogram: "<<n\*0.45359;

}

void gm\_kg ()

{

cout<<"Gram: ";

cin>>n;

cout<<"In Killogram: "<<n\*.001;

}

void kg\_gm ()

{

cout<<"Killogram: ";

cin>>n;

cout<<"In Gram: "<<n\*1000;

}

void kg\_pound ()

{

cout<<"Killogram: ";

cin>>n;

cout<<"In Pound: "<<2.20462\*n;

}

void gm\_pound ()

{

cout<<"Gram: ";

cin>>n;

cout<<"In Pound: "<<n\*.0022;

}

void pound\_gm ()

{

cout<<"Pound: ";

cin>>n;

cout<<"In Gram: "<<n\*453.59237;

}

void kg\_mton ()

{

cout<<"Killogram: ";

cin>>n;

cout<<"In MatricTon: "<<n\*.001;

}

void mton\_kg ()

{

cout<<"Matric ton: ";

cin>>n;

cout<<"In Killogram: "<<n\*1000;

}

};

class Temperature // Temperature Section

{

public:

double i;

void cel\_f ()

{

cout<<"Celsius: ";

cin>>i;

cout<<"In Fahrenheit: "<<(((9\*i)/5)+32);

}

void f\_cel ()

{

cout<<"Fahrenheit: ";

cin>>i;

cout<<"In Celsius: "<<(((i-32)/9)\*5);

}

void cel\_k ()

{

cout<<"Cellsius: ";

cin>>i;

cout<<"In Kelvin: "<<i+273;

}

void k\_cel ()

{

cout<<"Kelvin: ";

cin>>i;

cout<<"In Celsius: "<<i-273;

}

};

class Area // Area Section

{

public:

double s;

void mmSq\_cmSq ()

{

cout<<"Milli-Meter Square: ";

cin>>s;

cout<<"In Centi-Meter square: "<<s\*.01;

}

void cmSq\_mmSq ()

{

cout<<"Centi-Meter Square: ";

cin>>s;

cout<<"In Milli-Meter Square: "<<s\*100;

}

void cmSq\_mSq ()

{

cout<<"Cent-Meter Square: ";

cin>>s;

cout<<"In Meter Square: "<<s\*.0001;

}

void mSq\_cmSq ()

{

cout<<"Meter Square: ";

cin>>s;

cout<<"In Centi-Meter square: "<<s\*10000;

}

void mSq\_kmSq ()

{

cout<<"Meter Square: ";

cin>>s;

cout<<"In Killo-Meter Square: "<<s\*.000001;

}

void kmSq\_mSq ()

{

cout<<"Killo-Meter Square: ";

cin>>s;

cout<<"In Meter Square: "<<s\*1000000;

}

void feetSq\_mSq ()

{

cout<<"Sqaure Feet: ";

cin>>s;

cout<<"In Meter Square: "<<s\*.0929;

}

void mSq\_feetSq ()

{

cout<<"Meter Square: ";

cin>>s;

cout<<"In Square Feet: "<<s\*10.76391;

}

void yardSq\_mSq ()

{

cout<<"Yard Square: ";

cin>>s;

cout<<"In Meter Square: "<<s\*.83613;

}

void mSq\_yardSq ()

{

cout<<"Meter Square: ";

cin>>s;

cout<<"In Yard Square: "<<s\*1.19599;

}

void mileSq\_kmSq ()

{

cout<<"Mile Sqaure: ";

cin>>s;

cout<<"In Killo-Meter Square: "<<s\*2.5899;

}

void kmSq\_mileSq ()

{

cout<<"Killo-Meter Square: ";

cin>>s;

cout<<"In Mile Square: "<<s\*.3861;

}

void acre\_hect ()

{

cout<<"Acre: ";

cin>>s;

cout<<"In Hector: "<<s\*.40469;

}

void hect\_acre ()

{

cout<<"Hector: ";

cin>>s;

cout<<"In Acre: "<<s\*2.47105;

}

void acre\_kmSq ()

{

cout<<"Acre: ";

cin>>s;

cout<<"In Killo-Meter Square: "<<s\*.00405;

}

void kmSq\_acre ()

{

cout<<"Killo-Meter Sqaure: ";

cin>>s;

cout<<"In Acre: "<<s\*247.10538;

}

};

class Volume // Volume Section

{

public:

double p;

void mL\_L ()

{

cout<<"Milli-Litre: ";

cin>>p;

cout<<"In Litre: "<<p\*.001;

}

void L\_mL ()

{

cout<<"Litre: ";

cin>>p;

cout<<"In Milli-Litre: "<<p\*1000;

}

void cmQ\_mmQ ()

{

cout<<"Centi-Meter Cube: ";

cin>>p;

cout<<"In Milli-Meter Cube: "<<p\*1000;

}

void mmQ\_cmQ ()

{

cout<<"Milli-Meter Cube: ";

cin>>p;

cout<<"In Centi=Meter Cube: "<<p\*.001;

}

void cmQ\_mQ ()

{

cout<<"Centi-Meter Cube: ";

cin>>p;

cout<<"In Meter Cube: "<<p\*.000001;

}

void mQ\_cmQ ()

{

cout<<"Meter Cube: ";

cin>>p;

cout<<"In Centi-Meter Cube: "<<p\*1000000;

}

void inchQ\_mQ ()

{

cout<<"Cubic Inch: ";

cin>>p;

cout<<"In Meter Cube: "<<p\*.00002;

}

void mQ\_inchQ ()

{

cout<<"Meter Cube: ";

cin>>p;

cout<<"In Cubic Inch: "<<p\*61023.74409;

}

void feetQ\_mQ ()

{

cout<<"Cubic Feet: ";

cin>>p;

cout<<"In Meter Cube: "<<p\*1222 ;

}

void mQ\_feetQ ()

{

cout<<"Meter Cube: ";

cin>>p;

cout<<"In Cubic Feet: "<<p\*123;

}

void mQ\_gallon ()

{

cout<<"Meter Cube: ";

cin>>p;

cout<<"In Gallon: "<<p\*264.17205;

}

void gallon\_mQ ()

{

cout<<"Gallon: ";

cin>>p;

cout<<"In Meter Cube: "<<p\*.00379;

}

void L\_gallon ()

{

cout<<"Litre: ";

cin>>p;

cout<<"In Gallon: "<<p\*.21997;

}

void gallon\_L ()

{

cout<<"Gallon: ";

cin>>p;

cout<<"In Litre: "<<p\*4.54609;

}

};

int main () // Main Function

{

Length l;

Weight w;

Temperature t;

Volume v;

Area a;

int y;

char x,i;

string str = " UNIT CONVERSION CALCULATOR ";

int N = str.length();

HeadingString(str, N);

do

{

cout<<"\n TYPE"<<"\n ------";

cout<<"\n 1: Length\n" << " 2: Temparature\n"<<" 3: Weight\n"<<" 4: Area\n"<<" 5: Volume\n" <<" 6: Formulae Book\n";

cout<<"\n\nPlease choose your Convertion Type: ";

cin>>x;

if(x=='1')

{

cout<<"\n\n choose your unit convertion:\n";

cout<<"\n 1 : mm-m";

cout<<"\n 2 : m-mm";

cout<<"\n 3 : cm-m";

cout<<"\n 4 : m-cm";

cout<<"\n 5 : m-km";

cout<<"\n 6 : km-m";

cout<<"\n 7 : cm\_km";

cout<<"\n 8 : km-cm";

cout<<"\n 9 : m-mile";

cout<<"\n 10 : mile-m";

cout<<"\n 11 : km-mile";

cout<<"\n 12 : mile-km";

cout<<"\n 13 : feet-m";

cout<<"\n 14 : m-feet";

cout<<"\n 15 : inch-m";

cout<<"\n 16 : m-inch";

cout<<"\n 17 : yard-m";

cout<<"\n 18 : m-yard\n";

cout<<"choice: ";

cin>>y;

cout<<"----------------\n\n";

switch(y)

{

case 1:l.mm\_m();

break;

case 2:l.m\_mm();

break;

case 3:l.cm\_m();

break;

case 4:l.m\_cm();

break;

case 5:l.m\_km();

break;

case 6:l.km\_m();

break;

case 7:l.cm\_km();

break;

case 8:l.km\_cm();

break;

case 9:l.m\_mile();

break;

case 10:l.mile\_m();

break;

case 11:l.km\_mile();

break;

case 12:l.mile\_km();

break;

case 13:l.feet\_m();

break;

case 14:l.m\_feet();

break;

case 15:l.inch\_m();

break;

case 16:l.m\_inch();

break;

case 17:l.yard\_m();

break;

case 18:l.m\_yard();

break;

default:cout<<"YOU CHOSE WRONG OPTION!";

break;

}

}

else if(x=='2')

{

cout<<"\n\n choose your unit convertion:\n";

cout<<"\n 1: Celsius-Fahrenheit";

cout<<"\n 2: Fahrenheit-Celsius";

cout<<"\n 3: Celsius-Kelvin";

cout<<"\n 4: Kelvin-Celcius\n";

cout<<"Choice: ";

cin>>y;

cout<<"----------------\n\n";

switch(y)

{

case 1:t.cel\_f();

break;

case 2:t.f\_cel();

break;

case 3:t.cel\_k();

break;

case 4:t.k\_cel();

break;

default:cout<<"YOU CHOSE WRONG OPTION!";

break;

}

}

else if(x=='3')

{

cout<<"\n\nchoose your unit convertion :\n";

cout<<"\n 1: Milligm-Gramm";

cout<<"\n 2: Gramm-milligm";

cout<<"\n 3: Gramm-killogram";

cout<<"\n 4: killoGramm-Gramm";

cout<<"\n 5: pound-killogramm";

cout<<"\n 6: killogramm-pound";

cout<<"\n 7: Gramm-Pound";

cout<<"\n 8: Pound-gramm";

cout<<"\n 9: killogramm-Metric ton";

cout<<"\n 10: Metric ton-Killogramm\n";

cout<<"\nChoice: ";

cin>>y;

cout<<"----------------\n\n";

switch(y)

{

case 1:w.milligm\_gm();

break;

case 2:w.gm\_milligm();

break;

case 3:w.gm\_kg();

break;

case 4:w.kg\_gm();

break;

case 5:w.pound\_kg();

break;

case 6:w.kg\_pound();

break;

case 7:w.gm\_pound();

break;

case 8:w.pound\_gm();

break;

case 9:w.kg\_mton();

break;

case 10:w.mton\_kg();

break;

default: cout<<"YOU CHOSE WRONG OPTION!";

break;

}

}

else if(x=='4')

{

cout<<"\n\nchoose your unit convertion:\n";

cout<<"\n 1: Square mm-Square cm";

cout<<"\n 2: square cm-Square mm";

cout<<"\n 3: square cm-square m";

cout<<"\n 4: Square m-Square cm";

cout<<"\n 5: Square m-Square km";

cout<<"\n 6: Square km-Square m";

cout<<"\n 7: Square feet-Square m";

cout<<"\n 8: Square m-Square feet";

cout<<"\n 9: Square Yard-Square m";

cout<<"\n 10: Square m-Square yard";

cout<<"\n 11: Square mile-Square km";

cout<<"\n 12: Square km-Square mile";

cout<<"\n 13: Acre-Hectare";

cout<<"\n 14: Hectare-Acre";

cout<<"\n 15: Square km-Acre ";

cout<<"\n 16: Acre-Square km\n";

cout<<"Choice: ";

cin>>y;

cout<<"----------------\n\n";

switch(y)

{

case 1: a.mmSq\_cmSq();

break;

case 2: a.cmSq\_mmSq();

break;

case 3: a.cmSq\_mSq();

break;

case 4: a.mSq\_cmSq();

break;

case 5: a.mSq\_kmSq();

break;

case 6: a.kmSq\_mSq();

break;

case 7: a.feetSq\_mSq();

break;

case 8: a.mSq\_feetSq();

break;

case 9: a.yardSq\_mSq();

break;

case 10: a.mSq\_yardSq();

break;

case 11: a.mileSq\_kmSq();

break;

case 12: a.kmSq\_mileSq();

break;

case 13: a.acre\_hect();

break;

case 14: a.hect\_acre();

break;

case 15: a.kmSq\_acre();

break;

case 16: a.acre\_kmSq();

break;

default: cout<<"YOU CHOSE WRONG OPTION!";

break;

}

}

else if(x=='5')

{

cout<<"\n\nchoose your unit convertion:\n";

cout<<"\n 1 : ml-Litre";

cout<<"\n 2 : Litre-ml";

cout<<"\n 3 : Cubic mm-Cubic cm";

cout<<"\n 4 : Cubic cm-Cubic mm";

cout<<"\n 5 : Cubic cm-Cubic m";

cout<<"\n 6 : Cubic m-Cubic cm";

cout<<"\n 7 : Cubic Inch-Cubic m";

cout<<"\n 8 : Cubic m-Cubic Inch";

cout<<"\n 9 : Cubic feet-Cubic m";

cout<<"\n 10 : Cubic m-Cubic feet";

cout<<"\n 11 : Cubic m-Gallon (uk)";

cout<<"\n 12 : Gallon-Cubic m";

cout<<"\n 13 : Litre-Gallon";

cout<<"\n 14 : Gallon-Litre\n";

cout<<"Choise: ";

cin>>y;

cout<<"----------------\n\n";

switch(y)

{

case 1: v.mL\_L();

break;

case 2: v.L\_mL();

break;

case 3: v.mmQ\_cmQ();

break;

case 4: v.cmQ\_mmQ();

break;

case 5: v.cmQ\_mQ();

break;

case 6:

v.mQ\_cmQ();

break;

case 7:

v.inchQ\_mQ();

break;

case 8:

v.mQ\_inchQ();

break;

case 9:

v.feetQ\_mQ();

break;

case 10:

v.mQ\_feetQ();

break;

case 11:

v.mQ\_gallon();

break;

case 12:

v.gallon\_mQ();

break;

case 13:

v.L\_gallon();

break;

case 14:

v.gallon\_L();

break;

default: cout<<"YOU CHOSE WRONG OPTION!";

break;

}

}

else if(x=='6') //This will open the formulae book: Using File Handling

{

ofstream Formuale\_Book;

Formuale\_Book.open ("FormulaeBook.txt");

Formuale\_Book << "SOME IMPORTANT CONVERSIONS:\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n"<<endl<<endl<<"VOLUME\n";

Formuale\_Book<<"1 milliliter = 0.001 liter\n1 centiliter = 0.01 liter\n1 deciliter = 0.1 liter\n1 decaliter = 10 liters\n1 hectoliter = 100 liters\n1 kiloliter = 1000 liters\n1 cubic inch = 1.639 × 10^–2 liters\n1 gallon = 3.785 liters\n1 cubic foot = 28.316 liters\n\n";

Formuale\_Book<< "LENGTH\n1 millimeter = 0.001 meter\n1 centimeter = 0.01 meter\n1 decimeter = 0.1 meter\n1 decameter = 10 meters\n1 hectometer = 100 meters\n1 kilometer = 1000 meters\n1 inch = 2.54 × 10^−2 meters\n1 foot = 0.3048 meters\n1 angstrom = 1 x 10^-10 meters\n1 fermi = 1 x 10^-15 meters\n1 light year = 0.946 × 10^16 meters\n1 mile = 1.609344 kms\n\n";

Formuale\_Book<<"MASS\n1 milligram = 0.001 gram\n1 centigram = 0.01 gram\n1 decigram = 0.1 gram\n1 decagram = 10 gram\n1 hectogram = 100 gram\n1 kilogram = 1000 grams\n1 stone = 6350.29 grams\n1 pound = 453.592 grams\n1 ounce = 28.3495 grams\n\n";

Formuale\_Book<< "TIME\n1 minute = 60 seconds\n1 hour = 60 minutes / 3600 seconds\n1 day = 24 hours\n1 week = 7 days\n1 year = 365 days";

Formuale\_Book.close();

}

cout<<"\n----------------";

cout<<"\n\nCompleted Successfully! \n\nPress Y(Yes to convert again) & N (NO to Exit)!\n";

cin>>i;

}

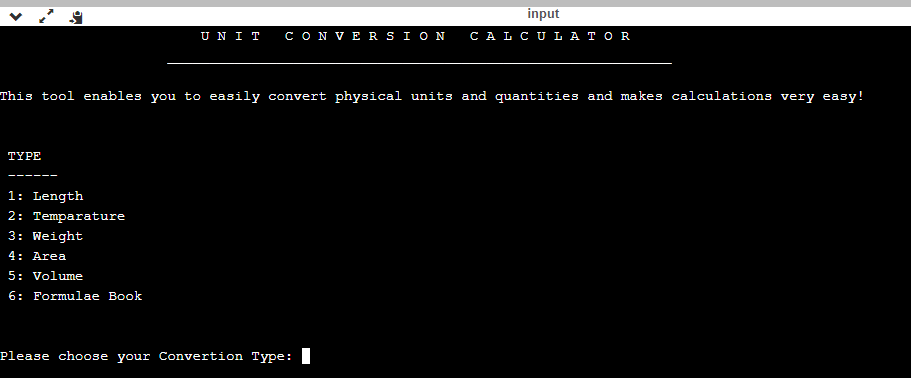
while(i!='n');

return 0;

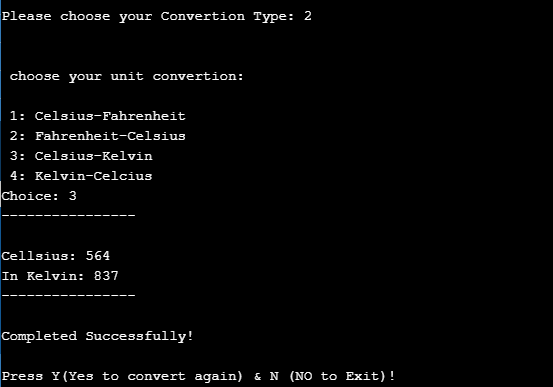
}

*Result and Discussion*

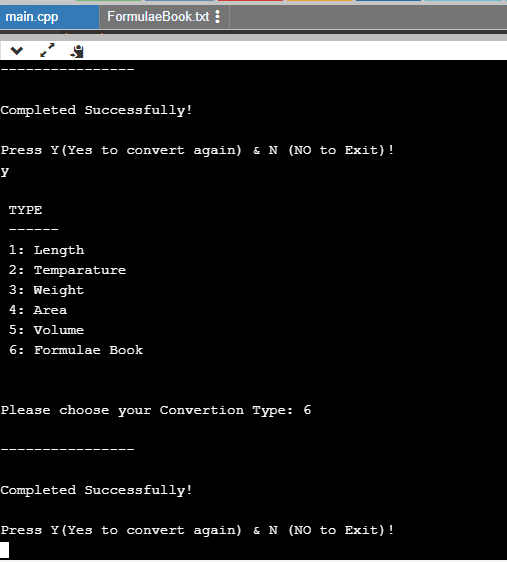
Overview of the output page



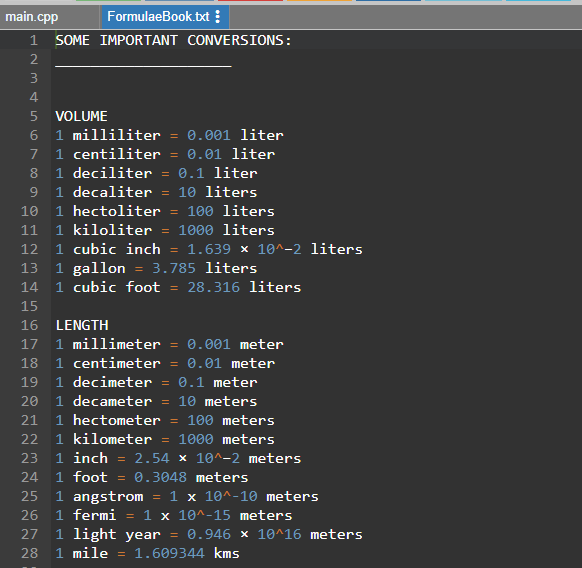
After choosing the appropriate quantity, choose the units among which the conversions have to be made



After successful conversion of units, the user may press Y to continue or N to terminate the program.



On continuing, if we further choose the option 6, a new .txt file created using file handling will be opened, containing necessary formulae used for carrying out conversions.



This is what the formulae book looks like

*Conclusion*

The end result of the calculator created during the course of this project is its ability to process different physical quantities that exist in this world and carry out conversions between commonly used units to provide adequate results.

This project will help immensely in the following way.

* It will make the calculations easier.
* It will save the user’s time spent on manual calculations.
* It will also be of a good assistance to any researcher.

With some modifications and addition of new features, this project can evolve as per the needs of the user in future.

*References*

<https://www.onlinegdb.com/online_c++_compiler>

<https://www.programiz.com/cpp-programming/online-compiler/>

<https://www.geeksforgeeks.org/iterate-over-characters-of-a-string-in-c/>

<https://www.geeksforgeeks.org/file-handling-c-classes/>