Week 8

EMT 101 – Engineering Programming

Dr. Farzad Ismail

School of Aerospace Engineering
Universiti Sains Malaysia
Nibong Tebal 14300 Pulau Pinang

Customized Library

- We have seen that using functions and subroutines defined outside of main to be efficient
- But having many functions or subroutines within the same main.cpp file can be difficult in terms of managing the size and understanding the main file.
- What about using subroutines and functions designed by other programmers in your own main file?
- This is where the use of customized library can be useful

How to create a customized library?

- Can be done by creating another (or many) .cpp files
- Within each .cpp file, you can include the functions and subroutines
- Since the *.cpp files are outside of main.cpp file, the functions and subroutines are not directly connected to main.cpp
- How to connect these additional *.cpp files to main.cpp?

Using Header file

- Similar to calling iostream library, you can connect these additional *.cpp files to main.cpp by using a header file and LINK all *.cpp files
- This header file will include ALL of the initializations of the functions and subroutines in each of the additional *.cpp files.
- If you are using a GUI-based compiler such as Dev C++, code block or Microsoft C++, you could LINK the *.cpp files by creating a project
- You need a link.bat to do the same on MS-DOS platform

Example

- Solving a fluid flow over an airfoil
- In a main function subfunctions or subroutines outside main program BUT still within the same main.cpp file
- Pseudo-code presentation

Functions within main.cpp

subroutine geometry and grid modelings

subroutine mathematical and physical models

Subroutine plot the velocity and pressure fields

```
int main()
   {  preprocessing steps
      ……..
      loop to solve the problem for pressure and velocities
      ……..
    return 0;
    }
```

Problems with all being in main.cpp

- Main.cpp becomes unnecessarily long and difficult to be understood
- Not flexible in terms of merging with codes done by other programmers (a standard practice in engineering)
- Difficult to extend the code for improvements and added capabilities

Main function + header file

```
#include <iostream>
#include “header_files.h”
int main()
   {   preprocessing steps

       loop to solve the problem for pressure &velocities

      return 0;
    }
```

Header file initializations

```
#include <iostream>
#include <cmath>
#define Pi 3.14159265358979
#define M 60
#define N 60
```

initializations of geometry and grid modelings subroutines

```
void Cell_Coord(double XCoord[M+2][N+2],
                double YCoord[M+2][N+2], variables);
```

initializations mathematical and physical models subroutines

Initializations of plot the velocity and pressure fields functions

Geometry.cpp file

```
#include <iostream>
#include <cmath>
# include “header_files.h” //if you need to access other
    functions outside geometry.cpp

void Cell_Coord(double XCoord[M+2][N+2],
                    double YCoord[M+2][N+2], variables)
{
  ……
}
void geometry ( ….)
{
}
```

Kinetic Energy Program

```
#include <iostream>
#include <string>
#include <cmath>
#include <conio.h>
#include “header_files.h”
using namespace std;

int main()
{   int n; double KE[n]; double u[n]; double v[n];
   double TKE;
   cout << "Enter number of particles:  " << endl;
   cin >>n;

  for (int i=0; i<n; i++)
      {
       cout << "Enter the u velocity for particles: " << endl;
       cin >>u[i];
       cout << "Enter the v velocity for particles: " << endl;
       cin >> v[i];
      }
 Determine_KE(n,u,v,KE, TKE);
getch();
return 0;
}
```

Header_files.h

```
#include <iostream>
#include <string>
#include <cmath>
#include <conio.h>
#define Pi  3.14159265358979
using namespace std;

//initialization of a subroutines to be used in Main or other *.cpp files

void Determine_KE(int, double u[], double v[], double KE[], double);
```

KE.cpp

```
#include <iostream>
#include <string>
#include <cmath>
#include <conio.h>
#include “header_files.h”
using namespace std;

//Declaration of a subroutine to be used in Main
void Determine_KE(int n, double u[], double v[], double KE[], double TKE)
 {
    for (int i=0; i<n; i++)
      {
       KE[i]= u[i]*u[i] + v[i]*v[i];
       TKE += KE[i];
      }
 cout  << "TKE is " << TKE << endl;
 }
```

Exercises

- Repeat Homework #2, but now please include

1. *subroutines to compute the second moment of inertia*
2. *Subroutines to compute the deflection*
3. *Subroutines to plot the deflection*

*in separate 3 *.cpp files connected within one project using header files.*