Week 4

EMT 101 – Engineering Programming

Dr. Farzad Ismail

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

Bugs in code

- There are three types of bugs (errors)
- Syntax errors – violation of the grammatical rules of the programming language
- A compiler would detect syntax errors.
- Semantic errors – violation of the ‘meaning’ or ‘action’ of the code – compiler does NOT detect and the code can run
- Algorithm errors – most difficult to be detected

Example: Syntax Error

```
int main ()
  {
   cout << “Hello world << endl;
   return 0;
  }
```

Syntax Error: undeclared identifier “cout”

Line 4 of program 1stprog.cpp

Example: Syntax Error

```
include <math>
int main ()
  {
   int num;
   float value;
   double bigNum;
   bignum = num + value;
  }
```

Can you detect the error?

Example: Semantic Error

```
char response;
cout << “Please input (y)es or (n)o: ” << endl;
cin >> response;
while ( (response != ‘y’) || (response!= ‘n’) )
      {
       cout << “Please try again. Enter (y)es or (n)o: ” << endl;
      }
```

The expression for while is always true regardless of what input you enter!

Discussion on semantic error example

- If user enters ‘y’, the first part of the expression is false but the second part is true -> overall true due to OR.
- If user enters ‘n’, the first part is true but second part is false -> true
- The program would keep on asking to try again regardless (infinite loop!)
- Corrected by (response != ‘y’) && (response!= ‘n’)

Example: Dangling if-else

-
  ```
  if (condition 1)
     if (condition 2)
       cout << “output:  ” << endl;
  else
       cout << “neither” << endl;
  ```

Correct version:

```
if (condition 1)
      {  if (condition 2)
         cout << “output:  ” << endl;
      }
 else
      cout << “neither” << endl;
```

Loops

- We have discussed previously on control structures (if-else).
- Now we present another programming tool: Loops
- Loops are used for iterative processes
- Very powerful tool in programming

Loops

- Do, while loops
- For loops
- Used to perform a repetitive (iterative) programming tasks
- Usually over some array

Example: Finding the Total Kinetic Energy

- Given a list of u and v velocity components, find the local KE
- Assume that u and v has n components
- How would you use for loop to calculate each ‘local’ KE?

Example: Finding the Local Kinetic Energy

```
for (int i=0; i < n; i++)
   {

     KE[i] = u[i]*u[i] + v[i]*v[i];

   }
```

Kinetic Energy Program

```
#include <iostream>
#include <string>
#include <cmath>
using namespace std;

int main()
{
  int n; double KE[n]; double u[n]; double v[n];
  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];
     }

 for (int i=0; i<n; i++)
    {
     cout << “Enter the v velocity for particles: “ << endl;
     cin >> v[i];
     }

 for (int i=0; i<n; i++)
    {
     KE[i]= u[i]*u[i] + v[i]*v[i];
     }
return 0;
}  // end of program body
```

Example: Finding the Total Kinetic Energy

- Now that you know each local KE, how can you calculate the total (global) KE?

-
  ```
  for (int i=0; i < n, i++)
    {
     KE[i] = u[i]*u[i] + v[i]*v[i];
    }
  double TKE = KE[0];
  ```

-
  ```
  for (int i=1; i < n, i++)
    {
     TKE =  KE[i] + TKE;
    }
  ```

Example: Finding the Total Kinetic Energy

- Notice that you have two for loops of the same size using the same information. Can we be more efficient?

```
double TKE = 0.0;
for (int i=0; i < n, i++)
   {
    KE[i] = u[i]*u[i] + v[i]*v[i];
    TKE += KE[i];
   }
```

Exercises

- Write a C++ program to solve an arbitrary matrix problem A= M*N where M and N are matrices in which you need to input the numbers on your screen. Assume M and N has a size 3 x 3.

- Write a program to perform a numerical integration of

  f(x)=x^2*sin(x)*exp(x^2) over x=[0,Pi].

  Use the simple rectangular area rule.

  Divide the domain into N subsections, where N=5,10,20,40. Compare your results.

Take Home: Exercises

- Write a C++ program to solve an arbitrary matrix problem A= M*N where M and N are matrices in which you need to input the numbers on your screen.

Now assume M and N has a size an arbitrary m x m size.

- Write a program to perform a numerical integration of

  f(x)=x^2*sin(x)*exp(x^2) over x=[0,Pi]. Use

  (i) the trapezoidal rule

  (ii) the Simpson's rule

  Divide the domain into N subsections, where N=5,10,20,40. Compare your results.