МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС

«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

Лабораторна робота № 4

з курсу «Проектування інформаційних систем»

Тема: «Модульне тестування (Unit-тести) та рефакторинг.»

\

Виконав:

студент IV курсу

групи ДА-32

Колінько Анжела

Київ – 2016

Мета: оволодіти навичками створення програмного забезпечення за методологією TDD та ознайомитися з процедурами рефакторинга.

1. Опис юніт тестів:

Розглянемо задачу перетворення футів і дюймів в метри і навпаки (як частину можливої системи переведення мір довжини).

Спочатку складемо unit-тести:

#include<iostream>

#include<math.h>

#include<time.h>

#include<cassert>

using namespace std;

#define ntst 10

#define eps 10e-5

class engm{

public:

int feet;

float inches;

bool operator==(engm eem){

if(eem.feet==feet&&fabs(eem.inches-inches)<eps) return true;

return false;

}

};

class dist{

private:

const float MTF;

float mtrs;

engm emtrs;

public:

dist(engm emu): emtrs(emu), MTF(3.280833F) {}

dist(float meters): mtrs(meters), MTF(3.280833F) {}

float em2m(){

return 0.;

}

engm m2em(){

engm emtmp;

return emtmp;

}

};

class tstdist{//1 feet is 3.280833F meter;

public:

static float mtf;

tstdist(){srand(time(NULL));}

void tstem2m(){

for(uint8\_t i=0;i<ntst;i++){

engm emtmp; emtmp.feet=rand()%200; emtmp.inches=rand()%12;//%200 to prevent overflow;

dist dst(emtmp);

float resm=(emtmp.feet+emtmp.inches/12.)/mtf;

assert(fabs(dst.em2m()-resm)<eps);

//cout<<'\n'<<dst.em2m()<<' '<<resm<<' '<<fabs(dst.em2m()-resm);

}

}

void tstm2em(){

for(uint8\_t i=0;i<ntst;i++){

float mtmp=rand()%200;//%200 to prevent overflow;

dist dst(mtmp);

engm resengm; resengm.feet=(int)(mtf\*mtmp); resengm.inches=12.\*(mtmp-resengm.feet);

//assert(dst.m2em()==resengm);

}

}

};

float tstdist::mtf=3.280833F;

int main(){

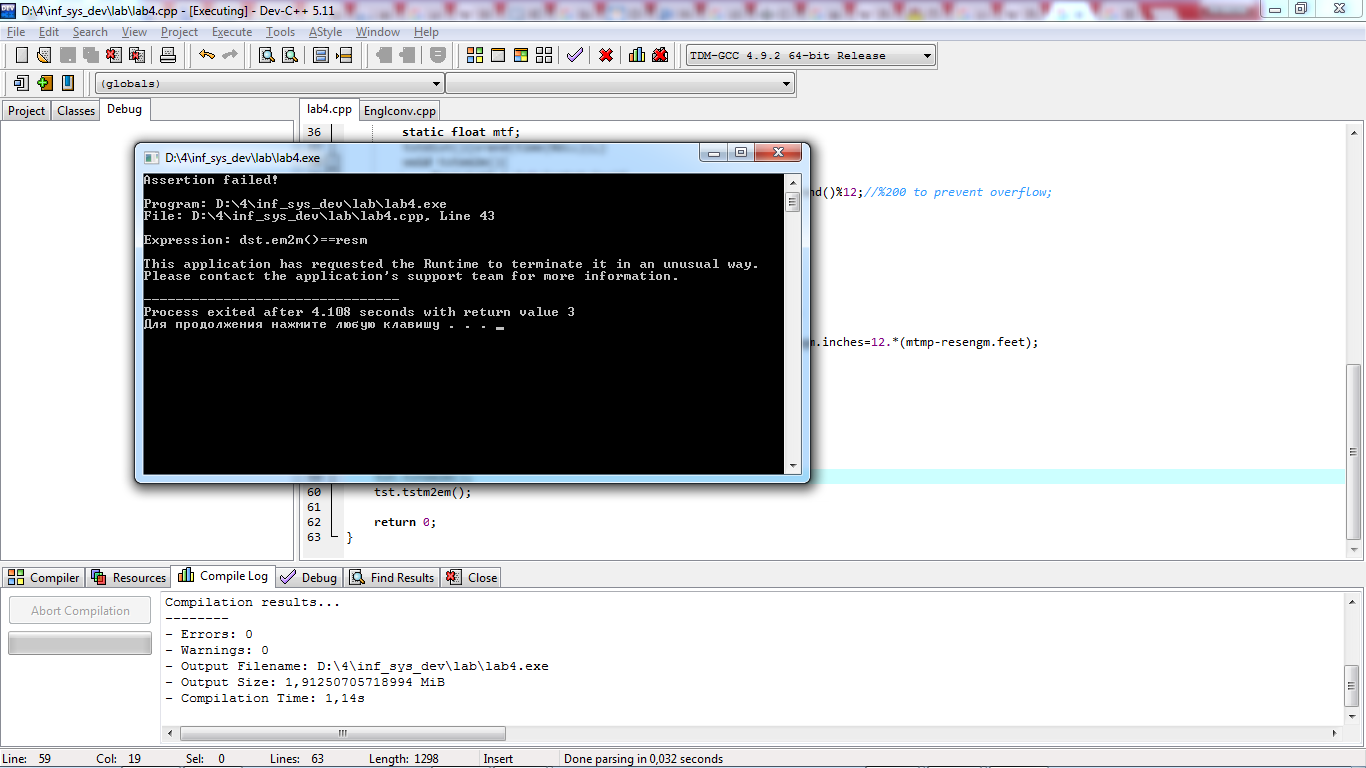
tstdist tst;

tst.tstem2m();

tst.tstm2em();

return 0;

}В результаті жоден тест не пройдений:



2. Спробуємо написати код для проходження unit-тестів:

#include<iostream>

#include<math.h>

#include<time.h>

#include<cassert>

using namespace std;

#define ntst 10

#define eps 10e-5

class engm{

public:

int feet;

float inches;

bool operator==(engm eem){

if(eem.feet==feet&&fabs(eem.inches-inches)<eps) return true;

cout<<"\neem.feet="<<eem.feet<<" feet="<<feet<<" eem.inches="<<eem.inches<<" inches="<<inches;

return false;

}

};

class dist{

private:

const float MTF;

float mtrs;

engm emtrs;

public:

dist(engm emu): emtrs(emu), MTF(3.280833F) {}

dist(float meters): mtrs(meters), MTF(3.280833F) {}

float em2m(){

float fracfeet=emtrs.inches/12.; //convert the inches

fracfeet += static\_cast<float>(emtrs.feet); //add the feet

return fracfeet/MTF;

}

engm m2em(){

engm emtmp;

float fltfeet=MTF \* mtrs; //convert to float feet

emtmp.feet=int(fltfeet); //feet is integer part

emtmp.inches=12\*(fltfeet-emtmp.feet);

return emtmp;

}

};

class tstdist{//1 feet is 3.280833F meter;

public:

static float mtf;

tstdist(){srand(time(NULL));}

void tstem2m(){

for(uint8\_t i=0;i<ntst;i++){

engm emtmp; emtmp.feet=rand()%200; emtmp.inches=rand()%12;//%200 to prevent overflow;

dist dst(emtmp);

float resm=(emtmp.feet+emtmp.inches/12.)/mtf;

assert(fabs(dst.em2m()-resm)<eps);

}

}

void tstm2em(){

for(uint8\_t i=0;i<ntst;i++){

float mtmp=rand()%200;//%200 to prevent overflow;

dist dst(mtmp);

engm resengm; resengm.feet=(int)(mtf\*mtmp); resengm.inches=12.\*(mtf\*mtmp-resengm.feet);

assert(dst.m2em()==resengm);

//cout<<(dst.m2em()==resengm)<<' ';

}

}

};

float tstdist::mtf=3.280833F;

int main(){

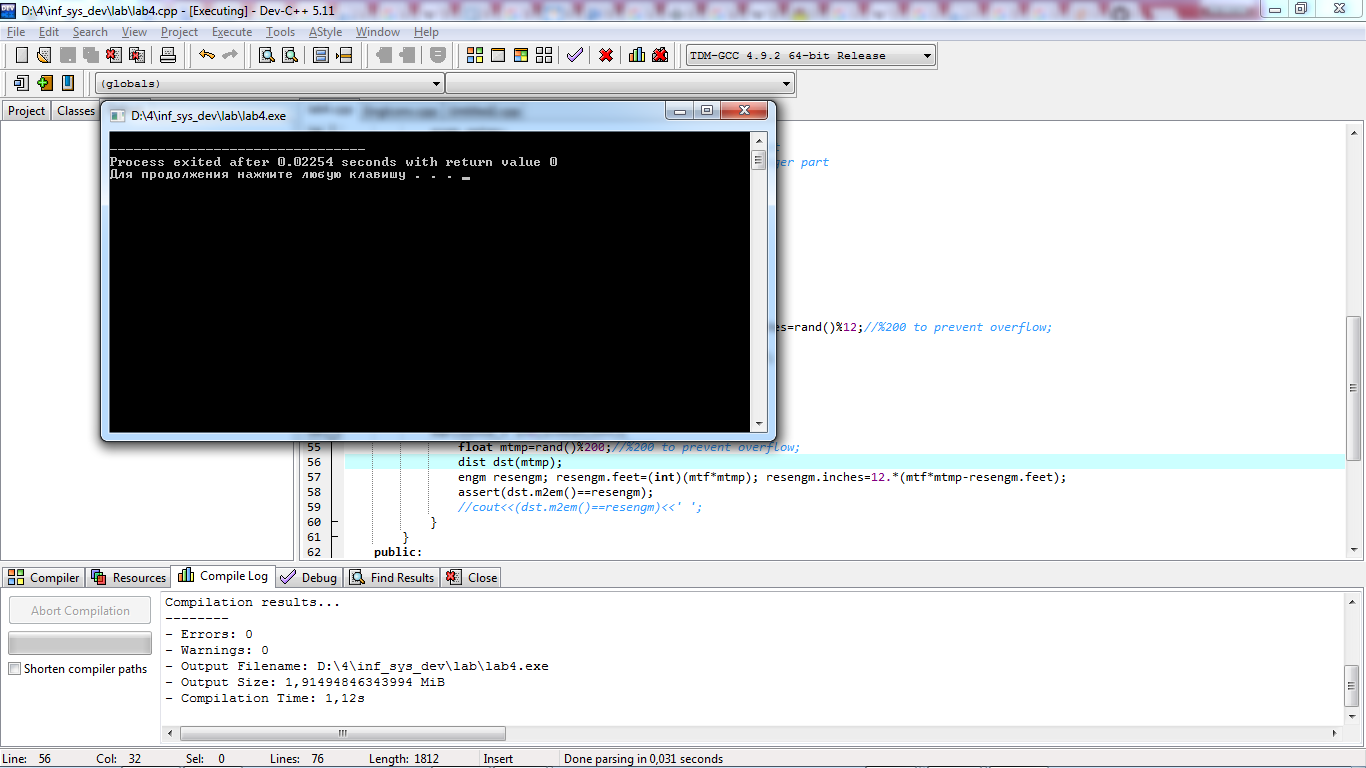
tstdist tst;

tst.tstem2m();

tst.tstm2em();

return 0;

}



3. Рефакторинг (частина коду зі змінами):

class tstdist{//1 feet is 3.280833F meter;

private:

static float mtf;

void tstem2m(){

for(uint8\_t i=0;i<ntst;i++){

engm emtmp; emtmp.feet=rand()%200; emtmp.inches=rand()%12;//%200 to prevent overflow;

dist dst(emtmp);

float resm=(emtmp.feet+emtmp.inches/12.)/mtf;

assert(fabs(dst.em2m()-resm)<eps);

}

}

void tstm2em(){

for(uint8\_t i=0;i<ntst;i++){

float mtmp=rand()%200;//%200 to prevent overflow;

dist dst(mtmp);

engm resengm; resengm.feet=(int)(mtf\*mtmp); resengm.inches=12.\*(mtf\*mtmp-resengm.feet);

assert(dst.m2em()==resengm);

//cout<<(dst.m2em()==resengm)<<' ';

}

}

public:

tstdist(){

srand(time(NULL));

tstem2m();

tstm2em();

}

};

float tstdist::mtf=3.280833F;

int main(){

tstdist tst;

return 0;

}

Висновок: на лабораторній роботі було розглянуто створення програмного забезпечення за методологією TDD.