МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС

«ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ»

Лабораторна робота № 5

з курсу «Проектування інформаційних систем»

Тема: «Система автоматичного створення довідника користувача та оформлення

коду за допомогою Coding Convention.»

Виконав:

студент IV курсу

групи ДА-32

Колінько Анжела

Київ – 2016

Мета: за допомогою системи генерації довідника користувача створити документ у форматі PDF і HTML для архітектурної програмної моделі.

1. Для оформлення коду була обрана High Integrity C++ Coding Standard:

Лістинг в додатках.

2. PDF документ згенерований doxygen:

За посиланням https://github.com/alex-and-ria/study/blob/inf\_sys\_dev/SF.pdf

Висновок: на лабораторній роботі було розглянуто роботу з системою генерації документації doxygen.

Додаток 1. Файл classes.h

//! \file \brief file with definitions

#ifndef CLASSES\_H

#define CLASSES\_H

#include<iostream>

using namespace std;

//! represents measurement system in feet and inches;

class engm{

public:

int feet;// \if you try making them private, be ready for rain dance with constructors;

float inches;// /

bool operator==(engm eem) const;//!< to compare two objects;

friend ostream& operator<<(ostream& os, const engm& em);//!< for enable using cout<<[engm\_object];

};

//! distance class which supports converting feet and inches into meters and vice versa;

class dist{

private:

const float MTF;

float mtrs;

engm emtrs;

bool ism;

public:

dist(engm emu);//!< constructor to set distance in feet and inches;

dist(float meters);//!< constructor to set distance in meters;

float em2m() const;//!<convertation feet and inches into meters;

engm m2em() const;//!<convertation merers into feet and inches;

friend ostream& operator<<(ostream& os, const dist& dst);//!< for enable using cout<<[dist\_object];

};

//! class to test methods in dist class

class tstdist{

private:

const static float mtf;

void tstem2m() const;

void tstm2em() const;

public:

tstdist();//!< contains void tstem2m() const; and void tstm2em() const;

friend ostream& operator<<(ostream&os, const tstdist& td);//!< for enable using cout<<[tstdist\_object];

};

#endif

Додаток 2. Файл classes.cpp

//! \file \brief realisation of methods; \details classes and their methods defined in classes.h;

#include<iostream>

#include<math.h>

#include<time.h>

#include<cassert>

#include"classes.h"

using namespace std;

#define ntst 10

#define eps 10e-5

//! \param eem is another object of engm with which comparison;

//!\return true if objects are equal (difference between inhes less than eps (#define eps 10e-5));

bool engm::operator==(engm eem) const{

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;

}

//! \param os[in] output stream, \param em[in] object, which fields are for outut;

//!\return os to enabe os<<[obj1]<<[obj2]<<[...];

ostream& operator<<(ostream& os, const engm& em){

os<<"\neem.feet="<<em.feet<<" eem.inches="<<em.inches;

return os;

}

dist::dist(engm emu): emtrs(emu), ism(0), MTF(3.280833F) {}

dist::dist(float meters): mtrs(meters), ism(1), MTF(3.280833F) {}

float dist::em2m() const{//!< \return distance in meters from distance in feet and inches which are fields of emtrs field of that object;

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

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

return fracfeet/MTF;

}

engm dist::m2em() const{//!< \return distance in feet and inches from distance in meters which is field mtrs of that object;

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;

}

ostream& operator<<(ostream& os, const dist& dst){//!< similary to operator<<(ostream& os, const engm& em);

if(dst.ism) {

os<<"\nmtrs="<<dst.mtrs; return os;

}

os<<"\ndst.emtrs.feet="<<dst.emtrs.feet<<" dst.emtrs.inches="<<dst.emtrs.inches;

return os;

}

//!1 feet is 3.280833F meter;

void tstdist::tstem2m() const{

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 tstdist::tstm2em() const{

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)<<' ';

}

}

tstdist::tstdist(){

srand(time(NULL));

tstem2m();

cout<<"testing em2m() complete";

tstm2em();

cout<<"\ntesting m2em() complete";

}

ostream& operator<<(ostream&os, const tstdist& td){//!< similary to operator<<(ostream& os, const engm& em);

cout<<"\ntesting em2m() complete";

cout<<"\ntesting m2em() complete";

}

const float tstdist::mtf=3.280833F;

Додаток 3. Файл mn.cpp

//! \file \brief main file to test methods in dist class;

#include<iostream>

#include"classes.cpp"

using namespace std;

int main(){

tstdist tst; /\*cout<<tst;

engm em; em.feet=5; em.inches=4; cout<<em;

dist dst(5); cout<<dst;

dist dst1(em); cout<<dst1;\*/

return 0;

}