

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ TECHNOLOGIJŲ KATEDRA

### ATOMINIŲ JĖGAINIŲ EFEKTYVUMO SKAIČIAVIMAS

#### Objektinio programavimo kursinis darbas

Darbą atliko: ISit-16 gr. Studentas

Ignas Sungaila

Darbą tikrino: lektorius Gintaras Dmitrijev

Vilnius, 2018

**Turinys**

[**Įvadas 3**](#_Toc523345181)

[**1. Bendroji dalis 4**](#_Toc523345182)

[**1.1. Uždavinio formulavimas 4**](#_Toc523345183)

[**1.2. Užduoties realizavimas 5**](#_Toc523345184)

[**1.2.1.** Užduoties realizacijos modelis, pavaizduotas supaprastinta UML klasių diagrama **5**](#_Toc523345185)

[**2. Programos realizavimo algoritmo aprašymas 6**](#_Toc523345186)

[**Rezultatai ir išvados 8**](#_Toc523345187)

[**Literatūra 9**](#_Toc523345188)

[**Priedai 10**](#_Toc523345189)

# **Įvadas**

Objektinis programavimas tai programavimo būdas skirtas suskirstyti projekto kodą taip, kad jis būtų paremtas objektais. Todėl, pasirinkta objektinio programavimo kursinio darbo tema – atominių jėgainių efektyvumo skaičiavimas. Minėta tema yra susijusi su fizikiniais ir matematiniais skaičiavimais remiantis „World Nuclear Association“ duomenimis, patalpintais internete. Surinkti duomenys, tokie kaip vidutiniškai per metus atominė jėgainė pagamina elektros energijos; kiek kiekviena pasirinkta jėgainė turi reaktorių, koks jų bendrasis galingumas ir šiluminė talpa.

Domintis šių dienų naujienomis galima pastebėti, jog elektros energijos gamybos procese dalyvauja ne tik atominės jėgainės, o ir tokios kaip vėjo, saulės, vandens, anglies, dujų jėgainės elektrai gaminti. Tačiau pastebima, jog apie branduolines elektros energijos gamybos jėgaines kalbama vis mažiau. Rašoma, jog aukščiau paminėtos atsinaujinančių energijos šaltinių jėgainės yra naudingesnės, mažiau teršiančios orą ir pan., tačiau branduolinė energija yra ypač svarbi užtikrinant pasaulyje augantį patikimos, prieinamos bei švarios energijos poreikį. Branduolinė energija yra žemų anglies emisijų į aplinką energija. Tai yra svarbus veiksnys sprendžiant klimato kaitos klausimus. Pagal tarpvyriausybinės klimato kaitos grupės (IPCC) ataskaitą (2014m.) norint realiai užtikrinti ne didesnį nei dviejų laipsnių globalų atšilimą iki 2050m. ne mažiau kaip 80% pasaulio elektros energijos gamybos turi būti pagaminama mažų anglies emisijų technologijomis. Galima paminėti ir tai, jog branduolinė energija turi daug daugiau gamybos privalumų, nei trūkumų lyginant su atsinaujinančių šalitinių elektros gamybos jėgainėmis.

**Programos tikslas** – palyginti 7 Pasaulio atominių jėgainių elektrines, kurios per metus vidutiniškai pagamina panašų elektros energijos kiekį. Jėgainės lyginamos apskaičiuojant jų efektyvumą bei suvartojamos materijos kiekį.

Programos kodas parašytas C++ kalba, „Microsoft Visual C++“ programavimo aplinkoje.

# **1. Bendroji dalis**

## **1.1. Uždavinio formulavimas**

„Atominių jėgainių efektyvumo skaičiavimas“ – kuriamos programos pavadinimas. Šią programą siekiama realizuoti tokiu modeliu:

Vartotojas, parašytame kode, sukuria atitinkamą klasės objektą ir jam priskiria atitinkamus duomenis. Veiksmas kartojamas tiek kartų, kiek norima lyginti atominių jėgainių. Taip pat atitinkamai kiekvienai branduolinei jėgainei sukuriami objektai, kurie aprašo veikiančius reaktorius. Paleidus programą ji apskaičiuoja atominių jėgainių efektyvumą bei materijos suvartojimo kiekį. Visa gauta informacija programoje pateikiama lentelės pavidalu. Gauti rezultatai gali padėti tobulinti elektros gamybos procesą ateityje bei proceso efektyvumą, kadangi elektros energijos pagaminimo kiekis pateiktose jėgainės yra panašus, tačiau nėra vienodas. Taip siekiama geresnių rezultatų.

**Užduotis.** Pasitelkus C++ kalbos Objektiniu programavimu, naudojant klases ir jų hierarchinę sistemą, sukurti tam tikro funkcionalumo teisingai veikiančią programą. Sukurta programa turi apskaičiuoti pateiktų branduolinių jėgainių efektyvumą elektros energijos gamybos procese bei suvartojamos materijos kiekį. Pradiniai duomenys, kurie yra pateikiami programoje, atitinkamai elektrinės pagaminamos vidutinės metinės elektros energijos kiekis, bendrasis galingumas, šiluminė talpa, veikiančių reaktorių skaičius, šalis ir branduolinės jėgainės pavadinimas. Rezultatai, kuriuos tikimasi gauti – atominės jėgainės efektyvumas, jos suvartojamos materijos kiekis ir apytikslis branduolinio kuro suvartojimo kiekis.

**Uždaviniai:**

**1.** Sudaryti atitinkamą klasių hierarchiją;

**2.** Kiekvieną klasę aprašyti aprašų (header) faile, o visus kiekvienos klasės metodus aprašyti jos realizacijos faile;

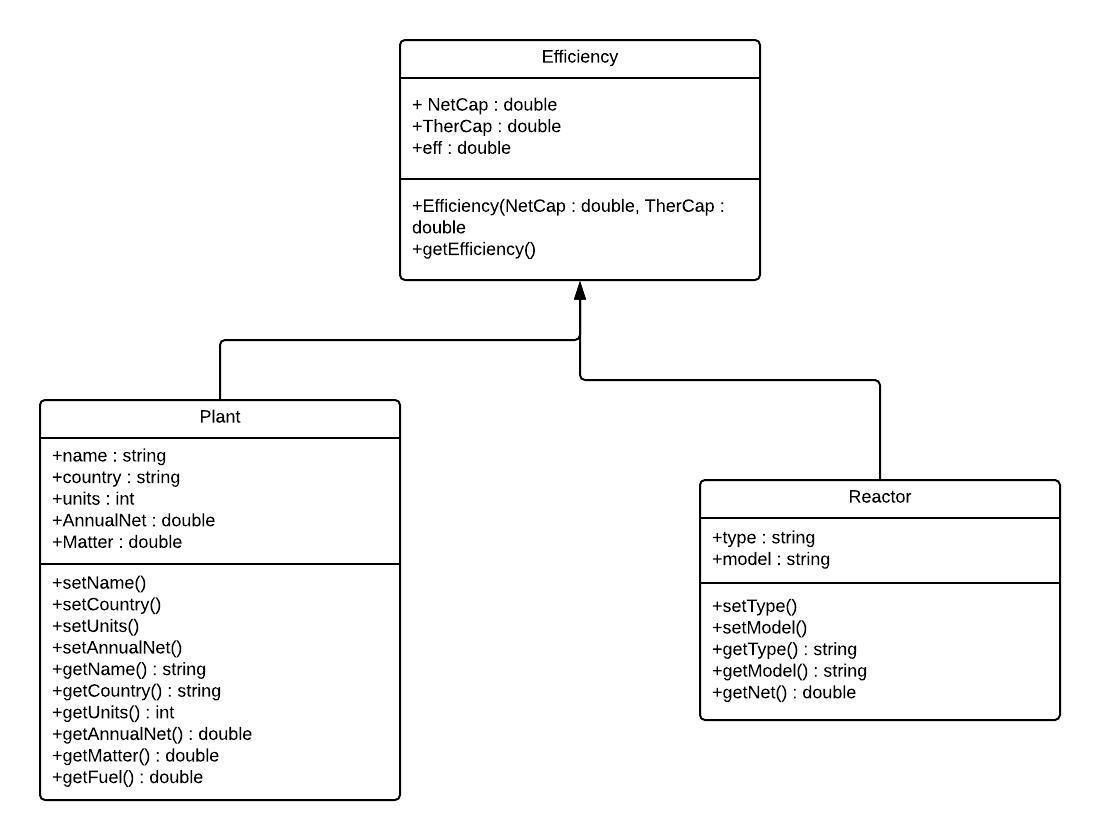
3. Pagrindiniame programos realizacijos faile sukurti objektus ir priskirti jiems atitinkamus duomenis;

4. Pagrindiniame programos realizacijos faile visus gaunamus talpinti į lentelę;

5. Realizuoti teisingai veikiančią programą.

## **1.2. Užduoties realizavimas**

### 1.2.1. Užduoties realizacijos modelis, pavaizduotas supaprastinta UML klasių diagrama

****

# **2. Programos realizavimo algoritmo aprašymas**

Pradedant kurti programą, kuri apskaičiuoja branduolinių jėgainių efektyvumą ir suvartojamą materijos kiekį, reikia sudaryti atitinkamą klasių hierarchiją. Mano parašytoje programoje naudosime dviejų skirtingų rūšių objektus – branduolinės jegainės ir branduolinių jėgainių reaktorius. Atitinkamai sukursime dvi skirtingas klases – vieną, kuri apibūdins atomines elektrines, o kitą kuri apibūdins jų reaktorius. Šios dvi skirtingos klasės turi vieną bendrą savybę – efektyvumą. Efektyvumui skaičiuoti sukuriama atskira klasė, kuri yra bazinė abiejoms iš ankščiau minėtų klasių.

Programos kodas pradedamas rašyti nuo klasių aprašymų. Pirma aprašoma bazinė klasė, skirta efektyvumui skaičiuoti, ji pavadinama ,,*Efficiency.h“*. Aprašų faile (*header)* iš anksto apibrėžiame reikšmė U = 0.0000433, kuri bus reikalinga tolimesniems skaičiavimams programoje. Taip pat apibrėžiame **protected** skiltyje šios klasės atributus *NetCap*  ir *TherCap,* kurie atitinkamai reiškia bendrajį galingumą ir šiluminę talpą. Skiltyje **public** sukuriame konstruktorių, kuriame bus talpinamos atributų reikšmės *Efficiency(NetCap, TherCap)* ir metodą *getEfficiency()* kuris bus skirtas efektyvumui skaičiuoti. Šios klasės realizacijos faile aprašome konstruktorių, priskirdami jam atributų reikšmes. Metodui, skirtam apskaičiuoti jegainių efektyvumą, pasinaudojame termodinamikos dėsniu, kuris teigia, kad visa sukurta šilumos energija, negali virsti elektros energija. Tuomet, metodą *getEfficiency()*, aprašome kaip grąžinantį reikšmę veiksmo: (*NetCap / TherCap) \* 100.* Šis veiksmas apskaičiuoja procentinę dalį: kiek elektros energijos pagamina jegainė iš sukurtos šilumos energijos.

Sekantis žingsnis – aprašyti Atominių jegainių ir reaktorių klases. Pradedame nuo atominių jegainių klasės kurią pavadiname ,,*Plant.h“* ir klasę *,,Efficiency.h“,* priskiriame kaip bazinę klasę. Skiltyje **private**apibrėžiame klasės atributus: jėgainės pavadinimą (*name),* šalį (*country)*, veikiančių reaktoių skaičių (*units)* ir jos vidutinį metinį pagaminamą elektros kiekį (*AnnualNet).* Skiltyje **public** sukuriame konstruktorių, kuris paveldi visas klasės *Efficiency* sukurto konstruktoriaus savybes. Toliau sukuriame visus reikalingus metodus šiai klasei: *setName(), setCountry(), setUnits(), setAnnualNet(), getName(), getCountry(), getUnits(), getAnnualNet(), getMatter(), getFuel()*. Šios klasės realizacijos faile aprašome visus minėtus metodus. Metodus – *setName(), setCountry(), setUnits() ir setAnnualNet()* aprašome kaip nuskaitančius atitinkamų atributų reikšmes ir priskiriame jas prie sukurto objekto. Metodus – *getName(), getCountry(), getUnits(), getAnnualNet()* aprašome kaip grąžinančius atitinkamų atributų reikšmes. Metodas *getMatter()* skirtas apskaičiuoti branduolinių jegainių materijos suvartojamam kiekiui skaičiuoti. Pagal *World Nuclear Association* pateiktus duomenis, jų internetiniame portale, veikiant 100% efektyvumui, reikia išdeginti 0,0000433 gramus Uraniumo išgrynintos materijos norint gauti 1kWh elektros energijos. Todėl pasinaudojus jegainių efektyvumu ir jų vidutiniu metiniu pagamintos elektros energijos kiekiu sukuriama formulė, kuri apskaičiuoja vidutinį metinį materijos suvartojimo kiekį. Ši formulė ir metodo *getMatter()* aprašymas pateikiamas štai taip: vidutinis metinis elektros energijos suvartojamas kiekis \* (0,0000433/ (efektyvumas / 100). Paskutinis šios klasės metodas *getFuel()* skirtas realesniam suvartojamo kuro kiekio įsivaizdavimui. Taip pat pagal *World Nuclear Association* duomenis, apytiksliai iš 250 tonų branduolinio kuro yra išgaunamas tik 1 kilogramas išgrynintos materijos kuri ir suteikia elektros energijos. *getFuel()* metodą aprašome paprastai materijos kiekį padauginę iš 250.

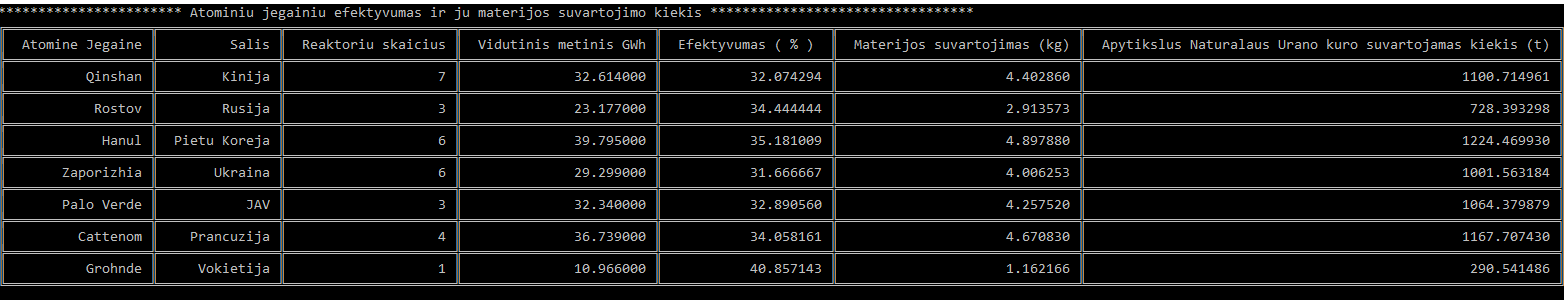
Kitas žingsnis yra aprašoma reaktorių klasė kurią pavadiname ,,*Reactor.h“*  ir klasę ,,*Efficiency“*  priskiriame kaip bazinę. Skiltyje **private** apibrėžiame atributus kurie nusako reaktoriaus tipą (*type*) ir modelį (*model*). Skiltyje **public** sukuriame konstruktorių kuris paveldi klasės *Efficiency*  konstruktoriaus savybes ir sukuriame metodus – *setType(), setModel(), getType(), getModel() ir getNet()*. Šios klasės realizacijos faile aprašome viską kas sukurta aprašų faile **public** skiltyje. *setType() ir setModel()* metodus aprašome kaip nuskaitomus reikšmes ir jas priskiriamas prie atitinkamų atributų. *getType()*, *getModel() ir getNet()* aprašome kaip grąžinančius reikšmes atitinkamų atributų.

Sukūrus tinkamą klasių hierarchiją ir visas klases aprašius, pradedame viską realizuoti pagrindiniame programos realizacijos faile. Į pagrindinį realizacijos failą įtraukiame visus sukurtus klasių aprašus ir pradedame kurti visus norimus objektus. Kursime 7 skirtingus branduolinių jegainių objektus. Šešios iš pasirinktų jegainių bus tos, kurios vidutiniškai per metus pagamina panašų kiekį elektros energijos. Viena atominė jėgainė yra įtraukta, nes yra laikoma pačia efektyviausia šiuo metu pasaulyje esanti atominė elektrinė. Šių jegainių duomenis galite pamatyti pirmame rezultatų paveiksle. Toliau atitinkamai kuriami kiekvienos atominių jegainių reaktorių objektai, kurie parodys individualų kiekvieno reaktoriaus tipą, modelį, galingumą bei efektyvumą.

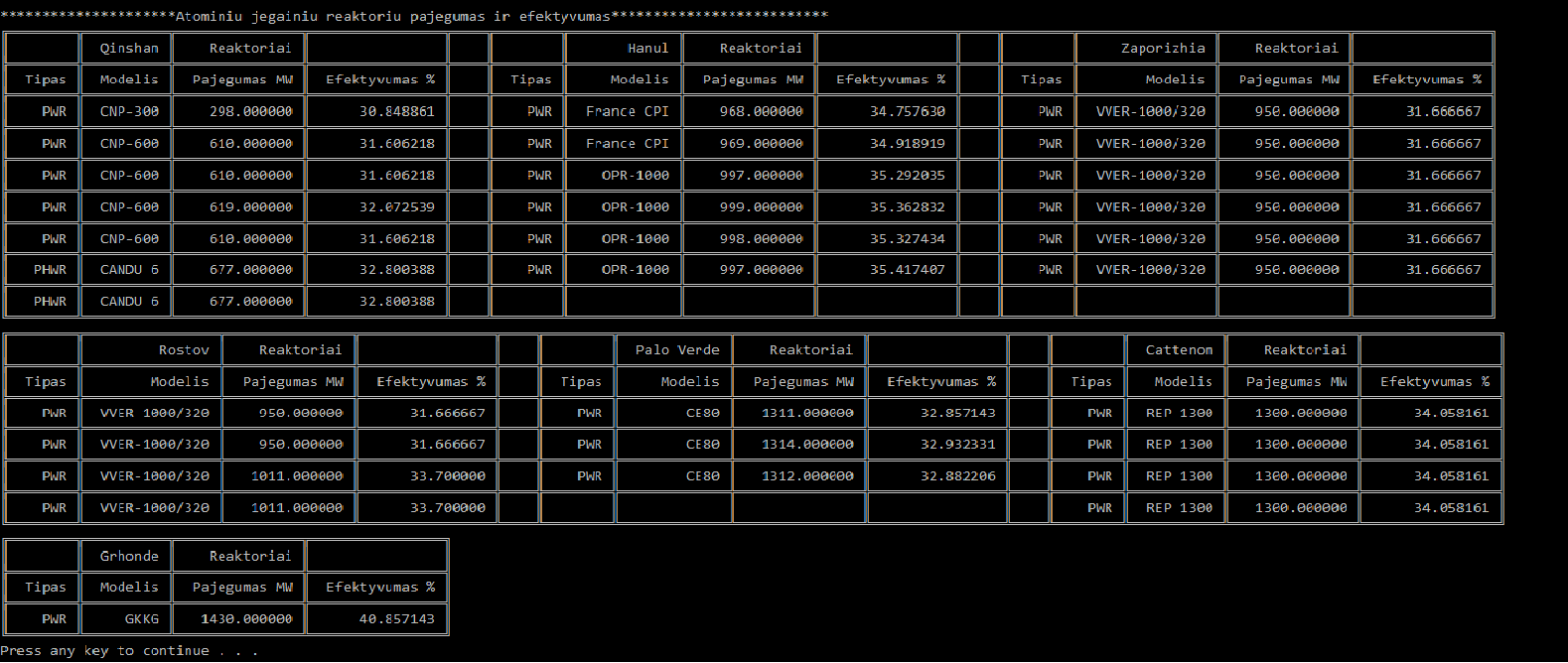
Paskutinis žingsnis realizuojant šią programą yra visų duomenų išdėstymas lentelės pavidalu. Tam, kad viskas būtų aiškiau išdėstyta naudojausi ,,*GitHub“* bendruomenės nario *Oradle* įrankiu ,,*ConsoleTable.cpp“* skirtu lentelės kūrimui C++ kalbos aplinkoje.

# **Rezultatai ir išvados**

Veikiančios programos pavyzdys nr. 1:

****

Iš pateikto veikiančios programos pavyzdžio galime pastebėti, jog programa yra nesudėtingos struktūros. Visi duomenys pateikiami lentelės pavidalu, kad būtų aiškiai suprantami ir galimi analizuoti. Lentelėje esantys duomenys parodo kiekvienos atominės elektrinės efektyvumą, todėl duomenis nėra sudėtinga palyginti. Matome, kad visos branduolinės jėgainės dirba palyginti mažu efektyvumu, kuris nesiekia nei 50%. O tai reiškia, kad atominės jėgainės gamindamos elektros energiją išeikvoja daugiau nei pusę visos gaminamos elektros energijos.

Veikiančios programos pavyzdys nr. 2:

Atominių jėgainių reaktorių pajėgumo ir efektyvumo lentelė iš pažiūros niekuo nesiskiria nuo pirmosios lentelės. Tačiau, šioje pateiktoje lentelėje matome individualių atominių jėgainių reaktorių pajėgumą ir efektyvumą.

Programa teisingai veikia siektu sukurti principu. Tačiau, kaip kiekviena programa turi privalumų, ši programa taip pat turi ir trūkumų, tokių kaip: duomenų trūkumas norint išanalizuoti šių dienų naujai kuriamus reaktorius.

# **Literatūra**

1. Narkevičius S. (2006). *Objektiškai orientuotas programavimas su C++: mokomoji knyga*. Vilniaus Universitetas Matematikos ir Informatikos fakultetas: Pavasaris. 44-60.

2. Lafore R. (2002). *Object – oriented Programming in C++, Fourth Edition*. Indiana: Sams Publishing. 10-15.

3. Vidžiūnas A. (2008). *C++ ir objektinis programavimas.* Smaltijos leidykla. 55-57.

4. Jusas V., Marcinkevičius R., Blonskis J., Bukšnaitis V., Rubliauskas D. (2016). *Objektinis programavimas ir dinaminiai sąrašai C++.* Kauno Technologijos Universitetas. 11-19.

5. World Nuclear Association. Nuoroda į internetą: <http://www.world-nuclear.org/> (žiūrėta 2018-08-15)

# **Priedai**

**Efficiency.h**

#ifndef Efficiency\_H

#define Efficiency\_H

#define U 0.0000433

/////bazine klase skirta atominiu jegainiu ir reaktoriu efektyvumui skaiciuoti///

class Efficiency

{

protected:

double NetCap, TherCap;

public:

Efficiency();

Efficiency(double NetCap, double TherCap);

virtual ~Efficiency();

double getEfficiency();

};

#endif

**Efficiency.cpp**

#include "pch.h"

#include "Efficiency.h"

Efficiency::Efficiency() //konstruktorius

{

}

Efficiency::~Efficiency() //destruktorius

{

}

Efficiency::Efficiency(double NetCap, double TherCap) {

this->NetCap = NetCap;

this->TherCap = TherCap;

}

double Efficiency::getEfficiency() { //funkcija skirta apskaiciuoti atomines jegaines efektyvumui

return (NetCap / TherCap) \* 100;

}

**Plant.h**

#ifndef Plant\_H

#define Plant\_H

#include "Efficiency.h"

#include <string>

///// atominiu jegainiu klase////

class Plant : public Efficiency

{

private:

std :: string name, country;

int units;

double AnnualNet;

public:

Plant();

~Plant();

Plant(double NetCap, double TherCap) : Efficiency::Efficiency(NetCap, TherCap) {};

void setName(std::string name);

void setCountry(std::string country);

void setUnits(int units);

void setAnnualNet(double AnnualNet);

std::string getName(void);

std::string getCountry(void);

int getUnits();

double getAnnualNet();

double getMatter();

double getFuel();

};

#endif

**Plant.cpp**

#include "pch.h"

#include "Plant.h"

Plant::Plant()

{

}

Plant::~Plant()

{

}

void Plant::setName(std::string name){

this->name = name;

}

void Plant::setCountry(std::string country) {

this->country = country;

}

void Plant::setUnits(int units) {

this->units = units;

}

void Plant::setAnnualNet(double AnnualNet) {

this->AnnualNet = AnnualNet;

}

std::string Plant::getName(void) {

return(name);

}

std::string Plant::getCountry(void) {

return(country);

}

int Plant::getUnits(void){

return(units);

}

double Plant::getAnnualNet(void){

return(AnnualNet);

}

double Plant::getMatter() {

return(getAnnualNet() \* pow(10, 6) \* (U / (getEfficiency() / 100))) / 1000;

}

double Plant::getFuel() {

return 250 \* getMatter();

}

**Reactor.h**

#ifndef Reactor\_H

#define Reactor\_H

#include "Efficiency.h"

#include <string>

/////// reaktoriu klase

class Reactor : public Efficiency

{

private:

std::string type, model;

public:

Reactor();

~Reactor();

Reactor(double NetCap, double TherCap):Efficiency::Efficiency(NetCap, TherCap) {};

void setType(std::string type);

std::string getType(void);

double getNet();

void setModel(std::string model);

std::string getModel(void);

};

#endif

**Reactor.cpp**

#include "pch.h"

#include "Reactor.h"

Reactor::Reactor()

{

}

Reactor::~Reactor()

{

}

void Reactor::setType(std::string type) {

this->type = type;

}

std::string Reactor::getType(void) {

return type;

}

double Reactor::getNet() {

return NetCap;

}

void Reactor::setModel(std::string model) {

this->model = model;

}

std::string Reactor::getModel(void) {

return model;

}

**Kursinis.cpp**

#include "pch.h"

#include <iostream>

#include "Efficiency.h"

#include "Plant.h"

#include <iomanip>

#include "CppConsoleTable.hpp"

#include "Reactor.h"

using ConsoleTable = samilton::ConsoleTable;

using namespace std;

int main()

{

//Atominiu jegainiu objektu kurimas//

Plant Qinshan(4110, 12814);

Qinshan.setName("Qinshan");

Qinshan.setCountry("Kinija");

Qinshan.setUnits(7);

Qinshan.setAnnualNet(32.614);

Plant Rostov(3100, 9000);

Rostov.setName("Rostov");

Rostov.setCountry("Rusija");

Rostov.setUnits(3);

Rostov.setAnnualNet(23.177);

Plant Hanul(5928, 16850);

Hanul.setName("Hanul");

Hanul.setCountry("Pietu Koreja");

Hanul.setUnits(6);

Hanul.setAnnualNet(39.795);

Plant Zaporizhia(5700, 18000);

Zaporizhia.setName("Zaporizhia");

Zaporizhia.setCountry("Ukraina");

Zaporizhia.setUnits(6);

Zaporizhia.setAnnualNet(29.299);

Plant PaloVerde(3937, 11970);

PaloVerde.setName("Palo Verde");

PaloVerde.setCountry("JAV");

PaloVerde.setUnits(3);

PaloVerde.setAnnualNet(32.340);

Plant Cattenom(5200, 15268);

Cattenom.setName("Cattenom");

Cattenom.setCountry("Prancuzija");

Cattenom.setUnits(4);

Cattenom.setAnnualNet(36.739);

Plant Grohnde(1430, 3500);

Grohnde.setName("Grohnde");

Grohnde.setCountry("Vokietija");

Grohnde.setUnits(1);

Grohnde.setAnnualNet(10.966);

///////Lenteles sudarymas////

ConsoleTable table(2, 1, ConsoleTable::Alignment::right);

table[0][0] = "Atomine Jegaine";

table[0][1] = "Salis";

table[0][2] = "Reaktoriu skaicius";

table[0][3] = "Vidutinis metinis GWh";

table[0][4] = "Efektyvumas ( % ) ";

table[0][5] = "Materijos suvartojimas (kg)";

table[0][6] = "Apytikslus Naturalaus Urano kuro suvartojamas kiekis (t)";

table[1][0] = Qinshan.getName();

table[1][1] = Qinshan.getCountry();

table[1][2] = Qinshan.getUnits();

table[1][3] = Qinshan.getAnnualNet();

table[1][4] = Qinshan.getEfficiency();

table[1][5] = Qinshan.getMatter();

table[1][6] = Qinshan.getFuel();

table[2][0] = Rostov.getName();

table[2][1] = Rostov.getCountry();

table[2][2] = Rostov.getUnits();

table[2][3] = Rostov.getAnnualNet();

table[2][4] = Rostov.getEfficiency();

table[2][5] = Rostov.getMatter();

table[2][6] = Rostov.getFuel();

table[3][0] = Hanul.getName();

table[3][1] = Hanul.getCountry();

table[3][2] = Hanul.getUnits();

table[3][3] = Hanul.getAnnualNet();

table[3][4] = Hanul.getEfficiency();

table[3][5] = Hanul.getMatter();

table[3][6] = Hanul.getFuel();

table[4][0] = Zaporizhia.getName();

table[4][1] = Zaporizhia.getCountry();

table[4][2] = Zaporizhia.getUnits();

table[4][3] = Zaporizhia.getAnnualNet();

table[4][4] = Zaporizhia.getEfficiency();

table[4][5] = Zaporizhia.getMatter();

table[4][6] = Zaporizhia.getFuel();

table[5][0] = PaloVerde.getName();

table[5][1] = PaloVerde.getCountry();

table[5][2] = PaloVerde.getUnits();

table[5][3] = PaloVerde.getAnnualNet();

table[5][4] = PaloVerde.getEfficiency();

table[5][5] = PaloVerde.getMatter();

table[5][6] = PaloVerde.getFuel();

table[6][0] = Cattenom.getName();

table[6][1] = Cattenom.getCountry();

table[6][2] = Cattenom.getUnits();

table[6][3] = Cattenom.getAnnualNet();

table[6][4] = Cattenom.getEfficiency();

table[6][5] = Cattenom.getMatter();

table[6][6] = Cattenom.getFuel();

table[7][0] = Grohnde.getName();

table[7][1] = Grohnde.getCountry();

table[7][2] = Grohnde.getUnits();

table[7][3] = Grohnde.getAnnualNet();

table[7][4] = Grohnde.getEfficiency();

table[7][5] = Grohnde.getMatter();

table[7][6] = Grohnde.getFuel();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Atominiu jegainiu efektyvumas ir ju materijos suvartojimo kiekis \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << table << endl;

//////////////Reaktorių objektų sudarymas///////

Reactor Q1(298, 966);

Q1.setType("PWR");

Q1.setModel("CNP-300");

Reactor Q2(610, 1930);

Q2.setType("PWR");

Q2.setModel("CNP-600");

Reactor Q3(610, 1930);

Q3.setType("PWR");

Q3.setModel("CNP-600");

Reactor Q4(619, 1930);

Q4.setType("PWR");

Q4.setModel("CNP-600");

Reactor Q5(610, 1930);

Q5.setType("PWR");

Q5.setModel("CNP-600");

Reactor Q6(677, 2064);

Q6.setType("PHWR");

Q6.setModel("CANDU 6");

Reactor Q7(677, 2064);

Q7.setType("PHWR");

Q7.setModel("CANDU 6");

Reactor R1(950, 3000);

R1.setType("PWR");

R1.setModel("VVER-1000/320");

Reactor R2(950, 3000);

R2.setType("PWR");

R2.setModel("VVER-1000/320");

Reactor R3(1011, 3000);

R3.setType("PWR");

R3.setModel("VVER-1000/320");

Reactor R4(1011, 3000);

R4.setType("PWR");

R4.setModel("VVER-1000/320");

Reactor H1(968, 2785);

H1.setType("PWR");

H1.setModel("France CPI");

Reactor H2(969, 2775);

H2.setType("PWR");

H2.setModel("France CPI");

Reactor H3(997, 2825);

H3.setType("PWR");

H3.setModel("OPR-1000");

Reactor H4(999, 2825);

H4.setType("PWR");

H4.setModel("OPR-1000");

Reactor H5(998, 2825);

H5.setType("PWR");

H5.setModel("OPR-1000");

Reactor H6(997, 2815);

H6.setType("PWR");

H6.setModel("OPR-1000");

Reactor P1(1311, 3990);

P1.setType("PWR");

P1.setModel("CE80");

Reactor P2(1314, 3990);

P2.setType("PWR");

P2.setModel("CE80");

Reactor P3(1312, 3990);

P3.setType("PWR");

P3.setModel("CE80");

Reactor G1(1430, 3500);

G1.setType("PWR");

G1.setModel("GKKG");

Reactor Z1(950, 3000);

Z1.setType("PWR");

Z1.setModel("VVER-1000/320");

Reactor Z2(950, 3000);

Z2.setType("PWR");

Z2.setModel("VVER-1000/320");

Reactor Z3(950, 3000);

Z3.setType("PWR");

Z3.setModel("VVER-1000/320");

Reactor Z4(950, 3000);

Z4.setType("PWR");

Z4.setModel("VVER-1000/320");

Reactor Z5(950, 3000);

Z5.setType("PWR");

Z5.setModel("VVER-1000/320");

Reactor Z6(950, 3000);

Z6.setType("PWR");

Z6.setModel("VVER-1000/320");

Reactor C1(1300, 3817);

C1.setType("PWR");

C1.setModel("REP 1300");

Reactor C2(1300, 3817);

C2.setType("PWR");

C2.setModel("REP 1300");

Reactor C3(1300, 3817);

C3.setType("PWR");

C3.setModel("REP 1300");

Reactor C4(1300, 3817);

C4.setType("PWR");

C4.setModel("REP 1300");

//////Reaktoriu talpinimas i lenteles////

ConsoleTable Rtable(2, 1, ConsoleTable::Alignment::right);

Rtable[0][1] = "Qinshan";

Rtable[0][2] = "Reaktoriai";

Rtable[1][0] = "Tipas";

Rtable[1][1] = "Modelis";

Rtable[1][2] = "Pajegumas MW";

Rtable[1][3] = "Efektyvumas %";

Rtable[2][0] = Q1.getType();

Rtable[2][1] = Q1.getModel();

Rtable[2][2] = Q1.getNet();

Rtable[2][3] = Q1.getEfficiency();

Rtable[3][0] = Q2.getType();

Rtable[3][1] = Q2.getModel();

Rtable[3][2] = Q2.getNet();

Rtable[3][3] = Q2.getEfficiency();

Rtable[4][0] = Q3.getType();

Rtable[4][1] = Q3.getModel();

Rtable[4][2] = Q3.getNet();

Rtable[4][3] = Q3.getEfficiency();

Rtable[5][0] = Q4.getType();

Rtable[5][1] = Q4.getModel();

Rtable[5][2] = Q4.getNet();

Rtable[5][3] = Q4.getEfficiency();

Rtable[6][0] = Q5.getType();

Rtable[6][1] = Q5.getModel();

Rtable[6][2] = Q5.getNet();

Rtable[6][3] = Q5.getEfficiency();

Rtable[7][0] = Q6.getType();

Rtable[7][1] = Q6.getModel();

Rtable[7][2] = Q6.getNet();

Rtable[7][3] = Q6.getEfficiency();

Rtable[8][0] = Q7.getType();

Rtable[8][1] = Q7.getModel();

Rtable[8][2] = Q7.getNet();

Rtable[8][3] = Q7.getEfficiency();

ConsoleTable Qtable(2, 1, ConsoleTable::Alignment::right);

Qtable[0][1] = "Rostov";

Qtable[0][2] = "Reaktoriai";

Qtable[1][0] = "Tipas";

Qtable[1][1] = "Modelis";

Qtable[1][2] = "Pajegumas MW";

Qtable[1][3] = "Efektyvumas %";

Qtable[2][0] = R1.getType();

Qtable[2][1] = R1.getModel();

Qtable[2][2] = R1.getNet();

Qtable[2][3] = R1.getEfficiency();

Qtable[3][0] = R2.getType();

Qtable[3][1] = R2.getModel();

Qtable[3][2] = R2.getNet();

Qtable[3][3] = R2.getEfficiency();

Qtable[4][0] = R3.getType();

Qtable[4][1] = R3.getModel();

Qtable[4][2] = R3.getNet();

Qtable[4][3] = R3.getEfficiency();

Qtable[5][0] = R4.getType();

Qtable[5][1] = R4.getModel();

Qtable[5][2] = R4.getNet();

Qtable[5][3] = R4.getEfficiency();

Rtable[0][6] = "Hanul";

Rtable[0][7] = "Reaktoriai";

Rtable[1][5] = "Tipas";

Rtable[1][6] = "Modelis";

Rtable[1][7] = "Pajegumas MW";

Rtable[1][8] = "Efektyvumas %";

Rtable[2][5] = H1.getType();

Rtable[2][6] = H1.getModel();

Rtable[2][7] = H1.getNet();

Rtable[2][8] = H1.getEfficiency();

Rtable[3][5] = H2.getType();

Rtable[3][6] = H2.getModel();

Rtable[3][7] = H2.getNet();

Rtable[3][8] = H2.getEfficiency();

Rtable[4][5] = H3.getType();

Rtable[4][6] = H3.getModel();

Rtable[4][7] = H3.getNet();

Rtable[4][8] = H3.getEfficiency();

Rtable[5][5] = H4.getType();

Rtable[5][6] = H4.getModel();

Rtable[5][7] = H4.getNet();

Rtable[5][8] = H4.getEfficiency();

Rtable[6][5] = H5.getType();

Rtable[6][6] = H5.getModel();

Rtable[6][7] = H5.getNet();

Rtable[6][8] = H5.getEfficiency();

Rtable[7][5] = H6.getType();

Rtable[7][6] = H6.getModel();

Rtable[7][7] = H6.getNet();

Rtable[7][8] = H6.getEfficiency();

Rtable[0][11]= "Zaporizhia";

Rtable[0][12]= "Reaktoriai";

Rtable[1][10]= "Tipas";

Rtable[1][11]= "Modelis";

Rtable[1][12]= "Pajegumas MW";

Rtable[1][13]= "Efektyvumas %";

Rtable[2][10]= Z1.getType();

Rtable[2][11]= Z1.getModel();

Rtable[2][12]= Z1.getNet();

Rtable[2][13]= Z1.getEfficiency();

Rtable[3][10] = Z2.getType();

Rtable[3][11] = Z2.getModel();

Rtable[3][12] = Z2.getNet();

Rtable[3][13] = Z2.getEfficiency();

Rtable[4][10] = Z3.getType();

Rtable[4][11] = Z3.getModel();

Rtable[4][12] = Z3.getNet();

Rtable[4][13] = Z3.getEfficiency();

Rtable[5][10] = Z4.getType();

Rtable[5][11] = Z4.getModel();

Rtable[5][12] = Z4.getNet();

Rtable[5][13] = Z4.getEfficiency();

Rtable[6][10] = Z5.getType();

Rtable[6][11] = Z5.getModel();

Rtable[6][12] = Z5.getNet();

Rtable[6][13] = Z5.getEfficiency();

Rtable[7][10] = Z6.getType();

Rtable[7][11] = Z6.getModel();

Rtable[7][12] = Z6.getNet();

Rtable[7][13] = Z6.getEfficiency();

Qtable[0][6] = "Palo Verde";

Qtable[0][7] = "Reaktoriai";

Qtable[1][5] = "Tipas";

Qtable[1][6] = "Modelis";

Qtable[1][7] = "Pajegumas MW";

Qtable[1][8] = "Efektyvumas %";

Qtable[2][5] = P1.getType();

Qtable[2][6] = P1.getModel();

Qtable[2][7] = P1.getNet();

Qtable[2][8] = P1.getEfficiency();

Qtable[3][5] = P2.getType();

Qtable[3][6] = P2.getModel();

Qtable[3][7] = P2.getNet();

Qtable[3][8] = P2.getEfficiency();

Qtable[4][5] = P3.getType();

Qtable[4][6] = P3.getModel();

Qtable[4][7] = P3.getNet();

Qtable[4][8] = P3.getEfficiency();

Qtable[0][11]= "Cattenom";

Qtable[0][12]= "Reaktoriai";

Qtable[1][10]= "Tipas";

Qtable[1][11]= "Modelis";

Qtable[1][12]= "Pajegumas MW";

Qtable[1][13]= "Efektyvumas %";

Qtable[2][10]= C1.getType();

Qtable[2][11]= C1.getModel();

Qtable[2][12]= C1.getNet();

Qtable[2][13]= C1.getEfficiency();

Qtable[3][10] = C2.getType();

Qtable[3][11] = C2.getModel();

Qtable[3][12] = C2.getNet();

Qtable[3][13] = C2.getEfficiency();

Qtable[4][10] = C3.getType();

Qtable[4][11] = C3.getModel();

Qtable[4][12] = C3.getNet();

Qtable[4][13] = C3.getEfficiency();

Qtable[5][10] = C4.getType();

Qtable[5][11] = C4.getModel();

Qtable[5][12] = C4.getNet();

Qtable[5][13] = C4.getEfficiency();

ConsoleTable Gtable(2, 1, ConsoleTable::Alignment::right);

Gtable[0][1] = "Grhonde";

Gtable[0][2] = "Reaktoriai";

Gtable[1][0] = "Tipas";

Gtable[1][1] = "Modelis";

Gtable[1][2] = "Pajegumas MW";

Gtable[1][3] = "Efektyvumas %";

Gtable[2][0] = G1.getType();

Gtable[2][1] = G1.getModel();

Gtable[2][2] = G1.getNet();

Gtable[2][3] = G1.getEfficiency();

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Atominiu jegainiu reaktoriu pajegumas ir efektyvumas\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << Rtable;

cout << Qtable;

cout << Gtable;

system("pause");

}