**«Язык программирования C++»**

**Тема: Патерны**

**Цель:**

Закрепить у слушателей практические навыки и теоретические знания для работы классами и объектами, свойствами. Научиться создавать иерархии классов.

**Задание 1.**

«Данные о системе обучения (Фабричный метод)»

Реализовать систему хранения информации о системе обучения. Информация опирается на следующие основные классы: Student, Teacher, Course. Эти классы связаны друг с другом посредством агрегации, ассоциации и проч.

Атрибуты классов

**student**

* id — уникальный численный идентификатор.
* name — имя.
* courses - курсы, которые посещает студент

**teacher**

* id — уникальный численный идентификатор.
* experience - стаж в годах
* name — имя
* courses - курсы, которые ведет преподаватель

**course**

* id — уникальный численный идентификатор.
* name - название курса
* teacher - id учителя, который ведет курс
* students - id студентов, посещающих курс

#include<iostream>

#include<string.h>

#include<windows.h>

#include<stdlib.h>

#include<fstream>

using namespace std;

class school

{

public:

virtual void doprogram() = 0;

};

class student : public school

{

protected:

int id;

string name;

string courses;

public:

student(int i, string n, string c)

{

id = i;

name = n;

courses = c;

}

void doprogram()

{

cout << "In progress" << endl;

}

};

class teacher : public school

{

protected:

int id;

int exp;

string name;

string courses;

public:

teacher(int i, int e, string n, string c)

{

id = i;

exp = e;

name = n;

courses = c;

}

void doprogram()

{

cout << "In progress" << endl;

}

};

class course : public school

{

protected:

int id;

string name;

int teacherid;

int studentid;

public:

course(int i, string n, int ti, int si)

{

id = i;

name = n;

teacherid = ti;

studentid = si;

}

void doprogram()

{

cout << "In progress" << endl;

}

};

class create

{

public:

virtual school\* crt1(int i, string n, string c) = 0;

virtual school\* crt2(int i, int e, string n, string c) = 0;

virtual school\* crt3(int i, string n, int ti, int si) = 0;

};

class studentcreate : public create

{

public:

school\* crt1(int i, string n, string c)

{

return new student(i, n, c);

}

school\* crt2(int i, int e, string n, string c)

{

return NULL;

}

school\* crt3(int i, string n, int ti, int si)

{

return NULL;

}

};

class teachercreate : public create

{

school\* crt1(int i, string n, string c)

{

return NULL;

}

school\* crt2(int i, int e, string n, string c)

{

return new teacher(i, e, n, c);

}

school\* crt3(int i, string n, int ti, int si)

{

return NULL;

}

};

class coursecreate : public create

{

school\* crt1(int i, string n, string c)

{

return NULL;

}

school\* crt3(int i, string n, int ti, int si)

{

return new course(i, n, ti, si);

}

school\* crt2(int i, int e, string n, string c)

{

return NULL;

}

};

int main()

{

create \*create1 = new studentcreate();

school\* student = create1->crt1(12321, "Vasya", "Programming");

create\* create2 = new teachercreate();

school\* teacher = create2->crt2(34346,15, "Bogdan", "Sys. administrating");

create\* create3 = new coursecreate();

school\* course = create3->crt3(34534,"Drawing", 345667, 456456);

}

**Задание 2.**

Задача «Данные о футбольном чемпионате (Фабричный метод)»

Реализовать систему хранения информации о футбольном чемпионате. Информация опирается на следующие основные классы: Team (команда), Player (игрок), Match (матч). Эти классы связаны друг с другом посредством агрегации, ассоциации и проч.

Атрибуты классов

Team

id — уникальный численный идентификатор.

name — имя.

players — игроки, играющие за данную команду в рамках чемпионата.

Player

id — уникальный численный идентификатор.

name — имя

team – команда.

Match

id — уникальный численный идентификатор.

date — дата.

location — место.

result — счёт.

team1 — первая команда.

team2 — вторая команда.

players – игроки, у частвовавшие в матче.

#include<iostream>

#include<string.h>

#include<windows.h>

#include<stdlib.h>

#include<fstream>

using namespace std;

class championship

{

public:

virtual void getinfo() = 0;

};

class team : public championship

{

private:

int id;

player players[11];

public:

team(int i, player\* arr)

{

id = i;

for (int i = 0; i < 11; i++)

{

players[i] = arr[i];

}

}

void getinfo()

{

cout << "id: " << id << endl;

for (int i = 0;i < 11; i++)

{

players[i].getinfo();

}

}

};

class player : public championship

{

private:

int id;

string name;

string team;

public:

player()

{

id = 0;

name = '\0';

team = '\0';

}

player(int i, string n, string t)

{

id = i;

name = n;

team = t;

}

void getinfo()

{

cout << "id: " << id << " name: " << name << " team: " << team << endl;

}

};

class match : public championship

{

private:

int id;

string date;

string loc;

string res;

string team1;

string team2;

player players[22];

public:

match(int i,string d, string l, string r, string t1, string t2, player \*p1, player \*p2)

{

id = i;

date = d;

loc = l;

res = r;

team1 = t1;

team2 = t2;

int j;

for (int i = 0; i < 22; i++)

{

if (i < 11)

{

players[i] = p1[i];

}

else

{

players[i] = p2[i - 11];

}

}

}

void getinfo()

{

cout << id << date << loc << res << team1 << team2 << endl;

for (int i = 0; i < 22; i++)

{

players[i].getinfo();

}

}

};

class create

{

public:

virtual championship\* crtplayer() = 0;

virtual championship\* crtteam() = 0;

virtual championship\* cermatch() = 0;

};

class createplayer : public create

{

public:

championship\* crtplayer(int i, string name, string team)

{

return new player(i, name, team);

}

championship\* crtteam()

{

return NULL;

}

championship\* cermatch()

{

return NULL;

}

};

class createteam : public create

{

public:

championship\* crtplayer()

{

return NULL;

}

championship\* crtteam(int i, player\* arr)

{

return new team(i, arr);

}

championship\* crtmatch()

{

return NULL;

}

};

class creatematch : public create

{

public:

championship\* crtplayer(int i, string name, string team)

{

return NULL;

}

championship\* crtteam()

{

return NULL;

}

championship\* crtmatch(int i, string d, string l, string r, string t1, string t2, player\* p1, player\* p2)

{

return new match(i, d, l, r, t1,t2,p1,p2);

}

};

int main()

{

championship\* players[11];

create\* crtpl = new createplayer();

for (int i = 0; i < 11; i++)

{

players[i] = crtpl->crtplayer(5677, "Vasya", "Dynamo");

}

create\* crttm = new createteam();

championship\* team = crttm->crtteam(454645, players);

create\* crtmtch = new creatematch();

championship\* match = crtmtch->cermatch(456456, "3/6/2006", "Ukraine", "2:0", "Dymano", "Shaktar", players, players);

}