# Жагсаалт (Лаборатори №10)

Багийн нэр: ЛОТ

Багийн ахлагч: З.Тунгалагмаа /20B1NUM2259/

Багийн гишүүд: О.Лхагвадулам /21B1NUM2670/

Б. Оюут /22B1NUM0514/

# ОРШИЛ

Энэхүү лабораторийн ажлаараа удамшлын харьцааны талаарх онолын судалгааг хийж, employee, person, spouse, child, division, jobdescription класуудыг тодорхойлж, тэдгээрийн харьцааг тодорхойлно.

# ЗОРИЛГО

Employee, person, spouse, child, division, jobdescription классуудыг байгуулжь удамшлын харьцааны талаарх ойлголтыг авах зорилготой. Үүний тулд дараах зорилтуудыг тавьж ажилласан:

1. Диаграммын дагуу классуудыг байгуулах,
2. Division болон JobDescription классуудын хэд хэдэн объект байгуулах,
3. Employee классын хэд хэдэн объект байгуулж тус бүрийн Division, JobDescription –ийг зааж өгөх
4. Employee классын объект тус бүрд Spouse, Child – уудыг тохируулах,
5. Employee классын объект тус бүрийн бүх мэдээллийг хэвлэх,
6. Харьцааг зөв тодорхойлох.

# ОНОЛЫН СУДАЛГАА

## 3.1 Загвар класс /template/

## Програмчлалын удамшил нь нэг класс өөр классаас шинж чанар, гишүүн функц удамших үед классуудын хоорондын харьцааг тогтоодог. Энэ харьцаа нь удамшлын замаар объект хандалгат програмчлалд хэрэгждэг. Aggregation, association, composition харьцаануудыг үүсгэлэг. Association харьцаа нь хоёр ба түүнээс дээш классын хоорондын харьцааг илэрхийдэг.

## 3.2 Жагсаалт

## Composition харьцаа нь нэг класс нь нөгөөгийнхөө бүрдэл хэсэг болдог төрлийн харьцаа. Энэ нь хэсэг нь бүхэлдээ байхгүй бол оршин тогтнох боломжгүй харилцааг илэрхийлдэг. Жишээлбэл, Машин нь жолооны хүрдтэй байдаг. Жолооны хүрд нь машины нэг хэсэг бөгөөд түүнгүйгээр оршин тогтнох боломжгүй юм. Хэрэв машин эвдэрсэн бол жолооны хүрд нь бас ажиллахгүй.

# ХЭРЭГЖҮҮЛЭЛТ

* Division болон JobDescription классуудын хэд хэдэн объект байгуулах,

// Division objects

Division division1, division2;

division1.setDivisionName("Sales");

division2.setDivisionName("Marketing");

//JobDescription objects

JobDescription jobDesc1, jobDesc2, jobDesc3;

jobDesc1.setDescription("Manager");

jobDesc2.setDescription("Engineer");

jobDesc3.setDescription("Teacher");

* Employee классын хэд хэдэн объект байгуулж тус бүрийн Division, JobDescription –ийг зааж өгөх

Employee классын параметрт байгуулагч функцэд division, jobdescription, spouse, child классуудын объектуудыг зааж өгсөн.

Employee employee1(division1, jobDesc1, spouse1, child1);

Employee employee2(division2, jobDesc2, spouse2, child2);

Employee employee3(division2, jobDesc3, spouse3, child1);

Employee(Division& division, JobDescription& jobDescription, Spouse& spouse, Child &child){

this->division=&division;

this->jobDescriptions.push\_back(&jobDescription);

this->spouse=&spouse;

this->children.push\_back(&child);

}

* Employee классын объект тус бүрд Spouse, Child – уудыг тохируулах,

//Spouse objects

Spouse spouse1, spouse2, spouse3;

spouse1.setName("Misheel");

spouse1.setAge(20);

spouse1.setAnniversaryDate(2020, 5, 11);

spouse2.setName("Enkhjin");

spouse2.setAge(30);

spouse3.setAnniversaryDate(2021, 7, 9);

spouse3.setName("Enerel");

spouse3.setAge(40);

spouse3.setAnniversaryDate(2019, 11, 7);

//Child objects

Child child1, child2, child3;

child1.setName("Baby1");

child1.setAge(5);

child1.setFavoriteToy("Doll");

child2.setName("Baby2");

child2.setAge(8);

child2.setFavoriteToy("Toy Car");

child3.setName("Baby3");

child3.setAge(2);

child3.setFavoriteToy("Teddy bear");

Employee(Division& division, JobDescription& jobDescription, Spouse& spouse, Child &child){

this->division=&division;

this->jobDescriptions.push\_back(&jobDescription);

this->spouse=&spouse;

this->children.push\_back(&child);

}

* Employee классын объект тус бүрийн бүх мэдээллийг хэвлэх,

cout << "Employee Information:" << endl;

cout << "---------------------" << endl;

// Employee 1

cout << "Name: " << employee1.getName() << endl;

cout << "SSN: " << employee1.getSSNum() << endl;

cout << "Age: " << employee1.getAge() << endl;

cout << "Company ID: " << employee1.getCompanyID() << endl;

cout << "Title: " << employee1.getTitle() << endl;

cout << "Division: " << employee1.division->getDivisionName() << endl;

cout << "Start Date: " << employee1.getStartDate().year << "-" << employee1.getStartDate().month << "-" << employee1.getStartDate().day << endl;

cout << "Spouse: " << employee1.spouse->getName() << endl;

cout << "Anniversary Date: " << employee1.spouse->getAnniversaryDate().year << "-" << employee1.spouse->getAnniversaryDate().month << "-" << employee1.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee1.children[0]->getName() << endl;

cout << "Child's Age: " << employee1.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee1.children[0]->getFavoriteToy() << endl;

// Employee 2

cout << "\nName: " << employee2.getName() << endl;

cout << "SSN: " << employee2.getSSNum() << endl;

cout << "Age: " << employee2.getAge() << endl;

cout << "Company ID: " << employee2.getCompanyID() << endl;

cout << "Title: " << employee2.getTitle() << endl;

cout << "Division: " << employee2.division->getDivisionName() << endl;

cout << "Start Date: " << employee2.getStartDate().year << "-" << employee2.getStartDate().month << "-" << employee2.getStartDate().day << endl;

cout << "Spouse: " << employee2.spouse->getName() << endl;

cout << "Anniversary Date: " << employee2.spouse->getAnniversaryDate().year << "-" << employee2.spouse->getAnniversaryDate().month << "-" << employee2.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee2.children[0]->getName() << endl;

cout << "Child's Age: " << employee2.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[0]->getFavoriteToy() << endl;

cout << "Child: " << employee2.children[1]->getName() << endl;

cout << "Child's Age: " << employee2.children[1]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[1]->getFavoriteToy() << endl;

cout << "Child: " << employee2.children[2]->getName() << endl;

cout << "Child's Age: " << employee2.children[2]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[2]->getFavoriteToy() << endl;

// Employee 3

cout << "\nName: " << employee3.getName() << endl;

cout << "SSN: " << employee3.getSSNum() << endl;

cout << "Age: " << employee3.getAge() << endl;

cout << "Company ID: " << employee3.getCompanyID() << endl;

cout << "Title: " << employee3.getTitle() << endl;

cout << "Division: " << employee3.division->getDivisionName() << endl;

cout << "Start Date: " << employee3.getStartDate().year << "-" << employee3.getStartDate().month << "-" << employee3.getStartDate().day << endl;

cout << "Spouse: " << employee3.spouse->getName() << endl;

cout << "Anniversary Date: " << employee3.spouse->getAnniversaryDate().year << "-" << employee3.spouse->getAnniversaryDate().month << "-" << employee3.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee3.children[0]->getName() << endl;

cout << "Child's Age: " << employee3.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee3.children[0]->getFavoriteToy() << endl;

* Харьцааг зөв тодорхойлох,

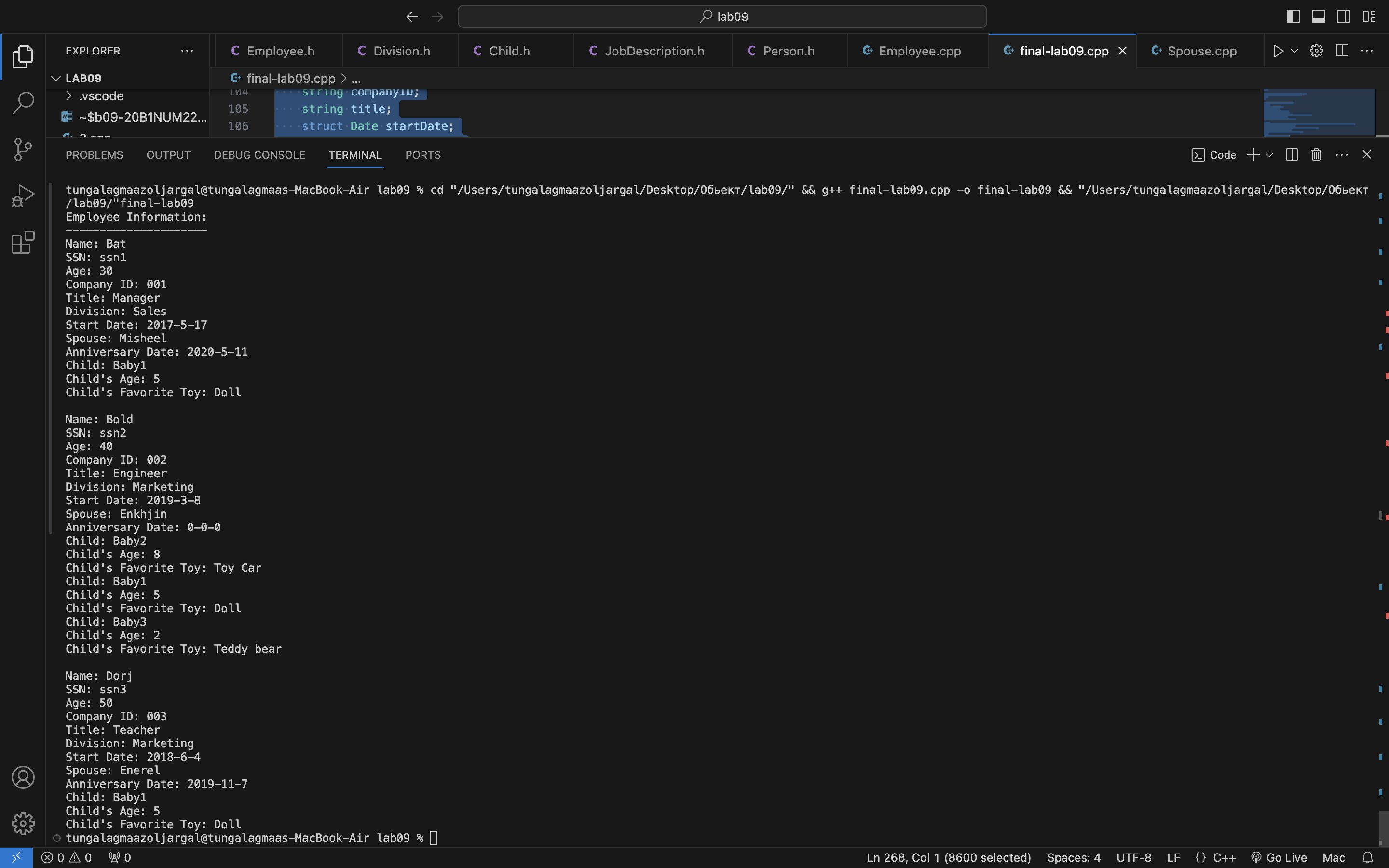
Division\* division; //1

vector<JobDescription\*> jobDescriptions;//1..n

Spouse\* spouse;//0..1

vector<Child\*> children;//0..n

Үр дүн:



# ДҮГНЭЛТ

Энэхүү лабораторийн ажилд C++ хэл дээрх объект хандалтат програмчлалын удамшлын харьцааны зарим ойлголтуудыг судалж, классуудад хэрэгжүүлсэн.

Програмчлалын удамшил нь нэг класс өөр классаас шинж чанар, гишүүн функц удамших үед классуудын хоорондын харьцааг тогтоодог. Энэ харьцаа нь удамшлын замаар объект хандалгат програмчлалд хэрэгждэг. Aggregation харьцаа нь тус тусдаа оршин тогтнох боломжтой харьцааг илэрхийлдэг бол бүрдэл харьцаа нь тус тусдаа оршин тогтнох боломжгүй харьцаа юм.

Харьцаа. Классууд хоорондын харьцааг заагч ашиглан тогтоосон. Employee нь хэлтэс, jobdescription, spouse, child-тай харьцаатай байхаар кодыг бичсэн. Division, jobdescription нь employee, spouse, child нь employee класстай холбоотой. Тогтоосон харьцааны дагуу (0..1, 0..n, ба 1..n) харилцааг хэрэгжүүлсэн. Жишээлбэл, employee нь spouse (0..1), тэг буюу түүнээс дээш child-тай (0..n), яг нэг division-тэй (1) байж болно. Харьцааг дараах байдлаар тодорхойлж өгдөг.

Division\* division; //1

vector<JobDescription\*> jobDescriptions;//1..n

Spouse\* spouse;//0..1

vector<Child\*> children;//0..n

# АШИГЛАСАН МАТЕРИАЛ

[1] Объект хандлагат технологийн С++ програмчлал, Ж.Пүрэв, 2008, Улаанбаатар.

# ХАВСРАЛТ

#include <iostream>

#include <string>

#include <vector>

using namespace std;

struct Date {

int year;

int month;

int day;

};

class Person {

public:

string name;

string SSNum;

int age;

public:

string getName() {

return name;

}

string getSSNum() {

return SSNum;

}

int getAge() {

return age;

}

void setName(string name) {

this->name = name;

}

void setSSNum(string SSNum) {

this->SSNum = SSNum;

}

void setAge(int age) {

this->age = age;

}

};

class Child : public Person {

public:

string favoriteToy;

public:

string getFavoriteToy() {

return favoriteToy;

}

void setFavoriteToy(string favoriteToy) {

this->favoriteToy = favoriteToy;

}

};

class Spouse : public Person {

public:

struct Date anniversaryDate;

public:

Date getAnniversaryDate() {

return anniversaryDate;

}

void setAnniversaryDate(int year, int month, int day) {

anniversaryDate.year = year;

anniversaryDate.month = month;

anniversaryDate.day = day;

}

};

class Division {

public:

string divisionName;

public:

string getDivisionName() {

return divisionName;

}

void setDivisionName(string divisionName) {

this->divisionName = divisionName;

}

};

class JobDescription {

public:

string description;

public:

string getDescription() {

return description;

}

void setDescription(string description) {

this->description = description;

}

};

class Employee : public Person {

public:

string companyID;

string title;

struct Date startDate;

Division\* division; //1

vector<JobDescription\*> jobDescriptions;//1..n

Spouse\* spouse;//0..1

vector<Child\*> children;//0..n

public:

Employee(Division& division, JobDescription& jobDescription, Spouse& spouse, Child &child){

this->division=&division;

this->jobDescriptions.push\_back(&jobDescription);

this->spouse=&spouse;

this->children.push\_back(&child);

}

string getCompanyID() {

return companyID;

}

string getTitle() {

return title;

}

Date getStartDate() {

return startDate;

}

void setCompanyID(string companyID) {

this->companyID = companyID;

}

void setTitle(string title) {

this->title = title;

}

void setStartDate(int year, int month, int day) {

startDate.year = year;

startDate.month = month;

startDate.day = day;

}

};

int main() {

struct Date startDate, anniversaryDate;

// Division objects

Division division1, division2;

division1.setDivisionName("Sales");

division2.setDivisionName("Marketing");

//JobDescription objects

JobDescription jobDesc1, jobDesc2, jobDesc3;

jobDesc1.setDescription("Manager");

jobDesc2.setDescription("Engineer");

jobDesc3.setDescription("Teacher");

//Spouse objects

Spouse spouse1, spouse2, spouse3;

spouse1.setName("Misheel");

spouse1.setAge(20);

spouse1.setAnniversaryDate(2020, 5, 11);

spouse2.setName("Enkhjin");

spouse2.setAge(30);

spouse3.setAnniversaryDate(2021, 7, 9);

spouse3.setName("Enerel");

spouse3.setAge(40);

spouse3.setAnniversaryDate(2019, 11, 7);

//Child objects

Child child1, child2, child3;

child1.setName("Baby1");

child1.setAge(5);

child1.setFavoriteToy("Doll");

child2.setName("Baby2");

child2.setAge(8);

child2.setFavoriteToy("Toy Car");

child3.setName("Baby3");

child3.setAge(2);

child3.setFavoriteToy("Teddy bear");

//Employee1 inputing all information

Employee employee1(division1, jobDesc1, spouse1, child1);

employee1.setName("Bat");

employee1.setSSNum("ssn1");

employee1.setAge(30);

employee1.setCompanyID("001");

employee1.setTitle("Manager");

employee1.setStartDate(2017, 5, 17);

//Employee2 inputing all information

Employee employee2(division2, jobDesc2, spouse2, child2);

employee2.setName("Bold");

employee2.setSSNum("ssn2");

employee2.setAge(40);

employee2.setCompanyID("002");

employee2.setTitle("Engineer");

employee2.setStartDate(2019, 3, 8);

employee2.children.push\_back(&child1);

employee2.children.push\_back(&child3);

//Employee3 inputing all information

Employee employee3(division2, jobDesc3, spouse3, child1);

employee3.setName("Dorj");

employee3.setSSNum("ssn3");

employee3.setAge(50);

employee3.setCompanyID("003");

employee3.setTitle("Teacher");

employee3.setStartDate(2018, 6, 4);

// information of all employees

cout << "Employee Information:" << endl;

cout << "---------------------" << endl;

// Employee 1

cout << "Name: " << employee1.getName() << endl;

cout << "SSN: " << employee1.getSSNum() << endl;

cout << "Age: " << employee1.getAge() << endl;

cout << "Company ID: " << employee1.getCompanyID() << endl;

cout << "Title: " << employee1.getTitle() << endl;

cout << "Division: " << employee1.division->getDivisionName() << endl;

cout << "Start Date: " << employee1.getStartDate().year << "-" << employee1.getStartDate().month << "-" << employee1.getStartDate().day << endl;

cout << "Spouse: " << employee1.spouse->getName() << endl;

cout << "Anniversary Date: " << employee1.spouse->getAnniversaryDate().year << "-" << employee1.spouse->getAnniversaryDate().month << "-" << employee1.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee1.children[0]->getName() << endl;

cout << "Child's Age: " << employee1.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee1.children[0]->getFavoriteToy() << endl;

// Employee 2

cout << "\nName: " << employee2.getName() << endl;

cout << "SSN: " << employee2.getSSNum() << endl;

cout << "Age: " << employee2.getAge() << endl;

cout << "Company ID: " << employee2.getCompanyID() << endl;

cout << "Title: " << employee2.getTitle() << endl;

cout << "Division: " << employee2.division->getDivisionName() << endl;

cout << "Start Date: " << employee2.getStartDate().year << "-" << employee2.getStartDate().month << "-" << employee2.getStartDate().day << endl;

cout << "Spouse: " << employee2.spouse->getName() << endl;

cout << "Anniversary Date: " << employee2.spouse->getAnniversaryDate().year << "-" << employee2.spouse->getAnniversaryDate().month << "-" << employee2.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee2.children[0]->getName() << endl;

cout << "Child's Age: " << employee2.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[0]->getFavoriteToy() << endl;

cout << "Child: " << employee2.children[1]->getName() << endl;

cout << "Child's Age: " << employee2.children[1]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[1]->getFavoriteToy() << endl;

cout << "Child: " << employee2.children[2]->getName() << endl;

cout << "Child's Age: " << employee2.children[2]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee2.children[2]->getFavoriteToy() << endl;

// Employee 3

cout << "\nName: " << employee3.getName() << endl;

cout << "SSN: " << employee3.getSSNum() << endl;

cout << "Age: " << employee3.getAge() << endl;

cout << "Company ID: " << employee3.getCompanyID() << endl;

cout << "Title: " << employee3.getTitle() << endl;

cout << "Division: " << employee3.division->getDivisionName() << endl;

cout << "Start Date: " << employee3.getStartDate().year << "-" << employee3.getStartDate().month << "-" << employee3.getStartDate().day << endl;

cout << "Spouse: " << employee3.spouse->getName() << endl;

cout << "Anniversary Date: " << employee3.spouse->getAnniversaryDate().year << "-" << employee3.spouse->getAnniversaryDate().month << "-" << employee3.spouse->getAnniversaryDate().day << endl;

cout << "Child: " << employee3.children[0]->getName() << endl;

cout << "Child's Age: " << employee3.children[0]->getAge() << endl;

cout << "Child's Favorite Toy: " << employee3.children[0]->getFavoriteToy() << endl;

return 0;

}