EOOP project

Author: Gaurav Chauhan

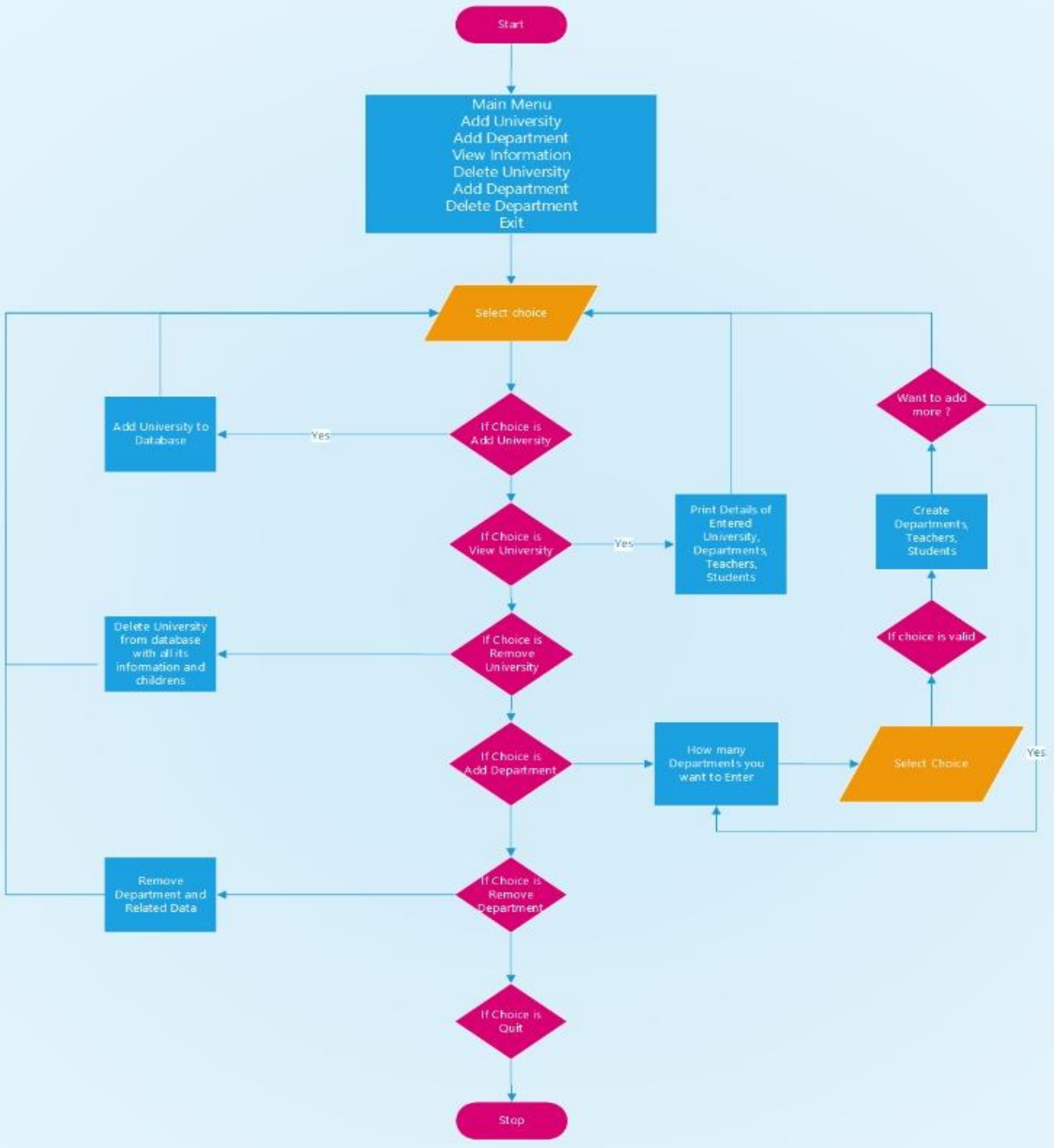
Key word: University

The Story:

This project allows for university administration to manage and keep track of a database of multiple universities and their departments with students and teachers. Each student is assigned a unique ID depending on the number of students ever employed by that university. Similarly, teachers are defined with unique ID as well. This allows for removal, addition of universities and departments with students and teachers. The format of the “constructors” allows for mass creation/deletion of student and teacher records. This also allows us to keep track of the departments each of the teachers and students they work/study in. in student we can provide each students full name with grades and date of birth and in teacher we can provide each teacher name with specialization. Each department is linked to only one specific university.

Case Study:

Relationship diagram:


Start
Main Menu
Add University
Add Department
View Information
Delete University
Add Department
Delete Department
Exit
Select choice
If Choice is Add University
Yes
Add University to Database
If Choice is View University
Yes
Print Details of Entered University, Departments, Teachers, Students
If Choice is Remove University
Delete University from database with all its information and childrens
If Choice is Add Department
How many Departments you want to Enter
Select Choice
If choice is valid
Create Departments, Teachers, Students
Want to add more ?
Yes
If Choice is Remove Department
Remove Department and Related Data
If Choice is Quit
Stop

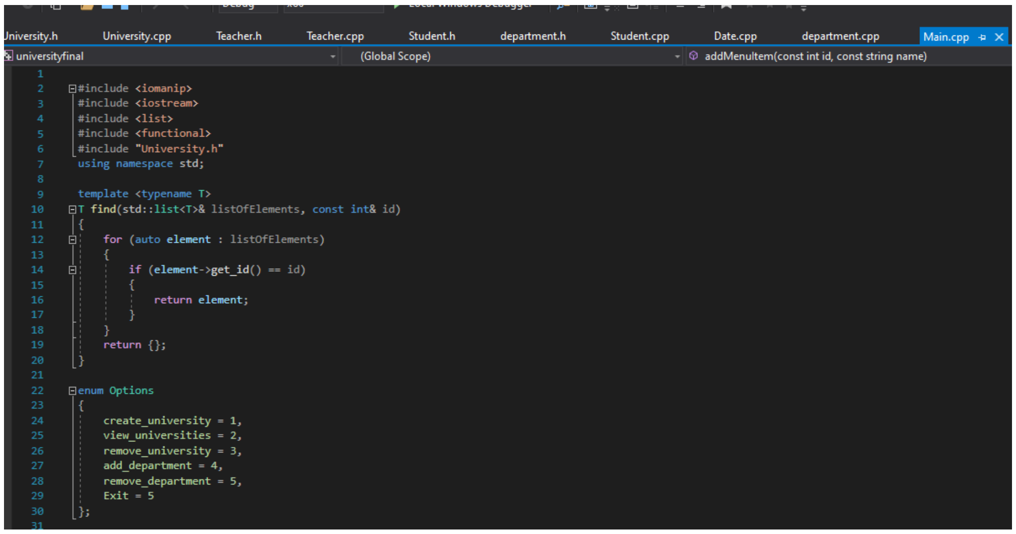
University class has pointers to departments, while each department has pointers to students and teachers.

Each university contains any number of departments, in each department we have any number of students and teachers in which students have their first name , last name , grades , date of birth and teacher have their first name , last name, specialization

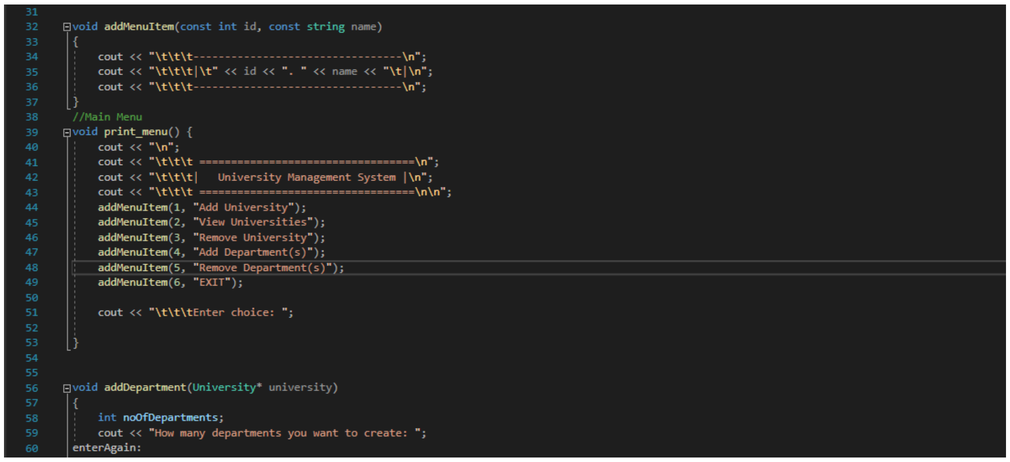
The Program:

The main class is the driver class from where all of the program is being executed, After there are others classes of University, Department, student and teacher .The University is the main class which contains multiple department list Department contains the list of students and teachers. Teacher and students classes are working independently where the adding and removing of students and teachers are done are respectively. Date classes is working as a helper to provide date to students(date of birth) and teachers .

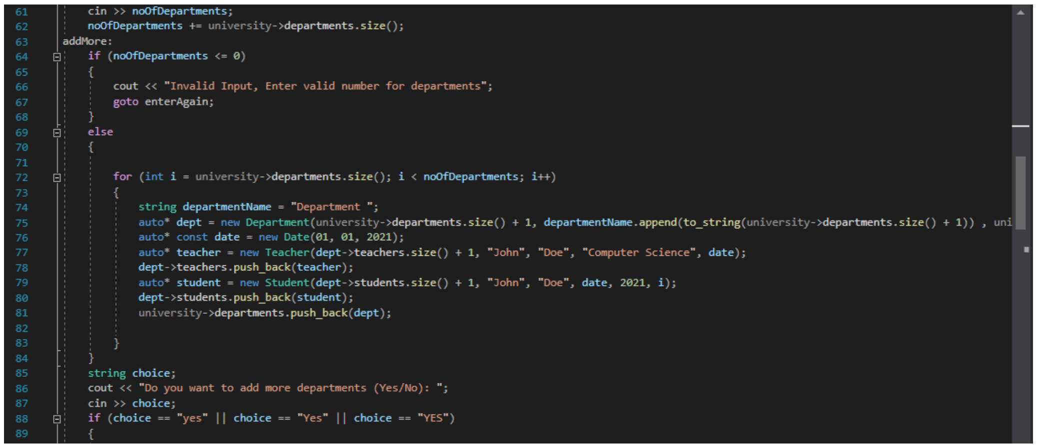
Main.cpp



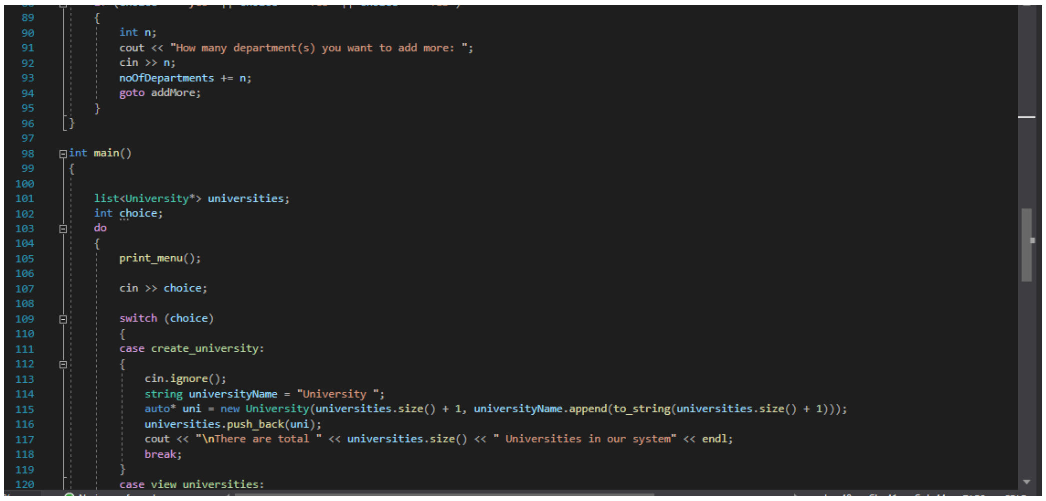
```cpp

#include <iomanip>
#include <iostream>
#include <list>
#include <functional>
#include "University.h"
using namespace std;

template <typename T>
T find(std::list<T>& listOfElements, const int& id)
{
    for (auto element : listOfElements)
    {
        if (element->get_id() == id)
        {
            return element;
        }
    }
    return {};
}

enum Options
{
    create_university = 1,
    view_universities = 2,
    remove_university = 3,
    add_department = 4,
    remove_department = 5,
    Exit = 5
};

```



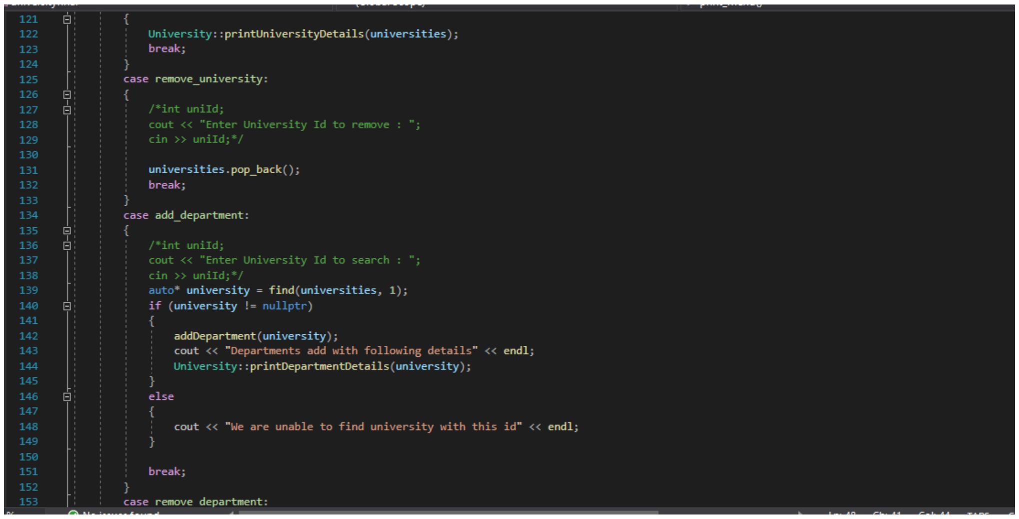
```cpp

void addMenuItem(const int id, const string name)
{
    cout << "\t\t\t---------------------------------\n";
    cout << "\t\t\t|\t" << id << ". " << name << "\t|\n";
    cout << "\t\t\t---------------------------------\n";
}
//Main Menu
void print_menu() {
    cout << "\n";
    cout << "\t\t\t ==================================\n";
    cout << "\t\t\t|   University Management System |\n";
    cout << "\t\t\t ==================================\n\n";
    addMenuItem(1, "Add University");
    addMenuItem(2, "View Universities");
    addMenuItem(3, "Remove University");
    addMenuItem(4, "Add Department(s)");
    addMenuItem(5, "Remove Department(s)");
    addMenuItem(6, "EXIT");

    cout << "\t\t\tEnter choice: ";

}


void addDepartment(University* university)
{
    int noOfDepartments;
    cout << "How many departments you want to create: ";
enterAgain:
```



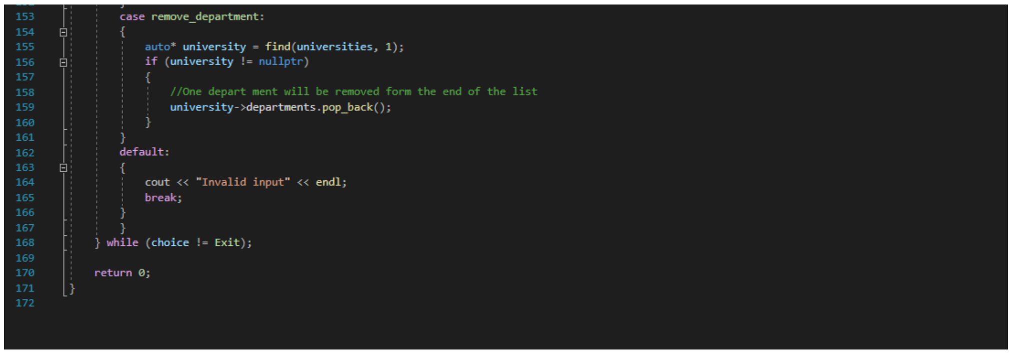
```cpp
    cin >> noOfDepartments;
    noOfDepartments += university->departments.size();
addMore:
    if (noOfDepartments <= 0)
    {
        cout << "Invalid Input, Enter valid number for departments";
        goto enterAgain;
    }
    else
    {

        for (int i = university->departments.size(); i < noOfDepartments; i++)
        {
            string departmentName = "Department ";
            auto* dept = new Department(university->departments.size() + 1, departmentName.append(to_string(university->departments.size() + 1)) , uni
            auto* const date = new Date(01, 01, 2021);
            auto* teacher = new Teacher(dept->teachers.size() + 1, "John", "Doe", "Computer Science", date);
            dept->teachers.push_back(teacher);
            auto* student = new Student(dept->students.size() + 1, "John", "Doe", date, 2021, i);
            dept->students.push_back(student);
            university->departments.push_back(dept);

        }
    }
    string choice;
    cout << "Do you want to add more departments (Yes/No): ";
    cin >> choice;
    if (choice == "yes" || choice == "Yes" || choice == "YES")
    {
```



```cpp
    {
        int n;
        cout << "How many department(s) you want to add more: ";
        cin >> n;
        noOfDepartments += n;
        goto addMore;
    }
}

int main()
{

    list<University*> universities;
    int choice;
    do
    {
        print_menu();

        cin >> choice;

        switch (choice)
        {
        case create_university:
        {
            cin.ignore();
            string universityName = "University ";
            auto* uni = new University(universities.size() + 1, universityName.append(to_string(universities.size() + 1)));
            universities.push_back(uni);
            cout << "\nThere are total " << universities.size() << " Universities in our system" << endl;
            break;
        }
        case view_universities:
```

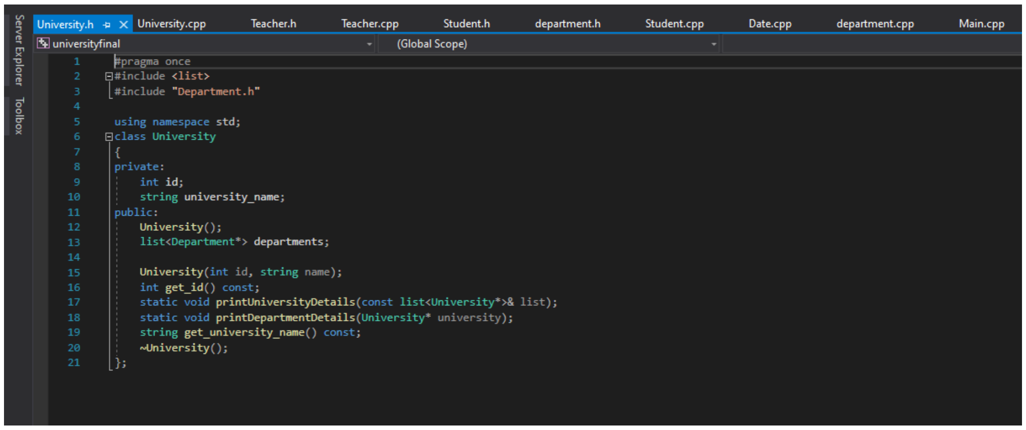


```cpp
        {
            University::printUniversityDetails(universities);
            break;
        }
        case remove_university:
        {
            /*int uniId;
            cout << "Enter University Id to remove : ";
            cin >> uniId;*/

            universities.pop_back();
            break;
        }
        case add_department:
        {
            /*int uniId;
            cout << "Enter University Id to search : ";
            cin >> uniId;*/
            auto* university = find(universities, 1);
            if (university != nullptr)
            {
                addDepartment(university);
                cout << "Departments add with following details" << endl;
                University::printDepartmentDetails(university);
            }
            else
            {
                cout << "We are unable to find university with this id" << endl;
            }

            break;
        }
        case remove_department:
```



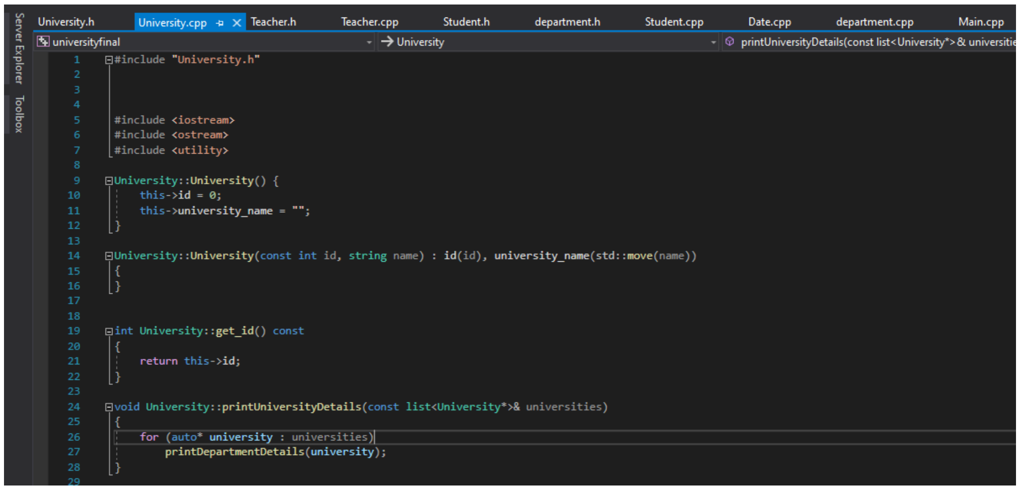
```cpp
        case remove_department:
        {
            auto* university = find(universities, 1);
            if (university != nullptr)
            {
                //One depart ment will be removed form the end of the list
                university->departments.pop_back();
            }
        }
        default:
        {
            cout << "Invalid input" << endl;
            break;
        }
        }
    } while (choice != Exit);

    return 0;
}
```

University.h

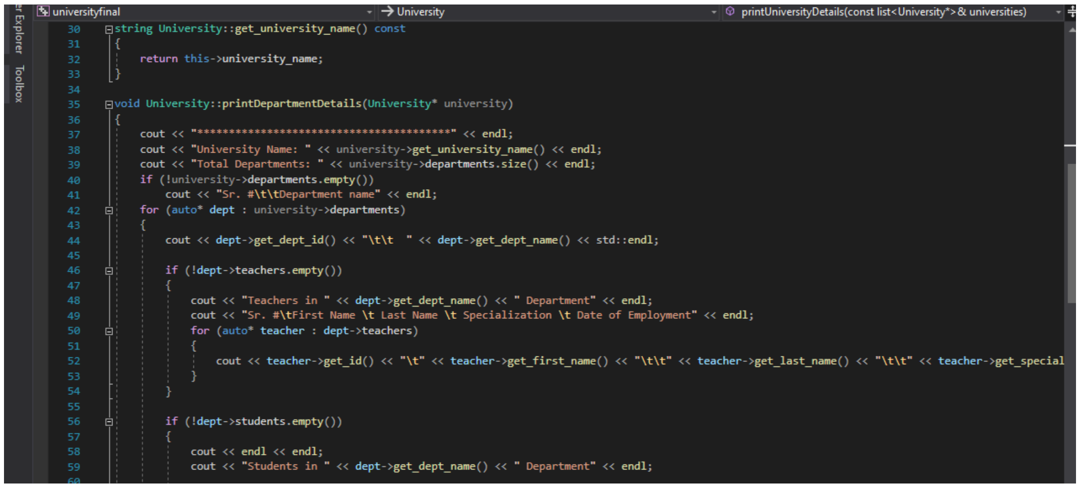


```cpp
#pragma once
#include <list>
#include "Department.h"

using namespace std;
class University
{
private:
    int id;
    string university_name;
public:
    University();
    list<Department*> departments;

    University(int id, string name);
    int get_id() const;
    static void printUniversityDetails(const list<University*>& list);
    static void printDepartmentDetails(University* university);
    string get_university_name() const;
    ~University();
};
```

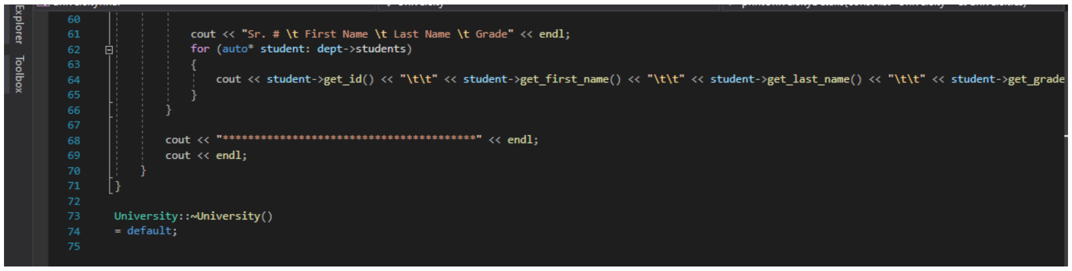
University.cpp



```cpp
#include "University.h"



#include <iostream>
#include <ostream>
#include <utility>

University::University() {
    this->id = 0;
    this->university_name = "";
}

University::University(const int id, string name) : id(id), university_name(std::move(name))
{
}


int University::get_id() const
{
    return this->id;
}

void University::printUniversityDetails(const list<University*>& universities)
{
    for (auto* university : universities)
        printDepartmentDetails(university);
}

```

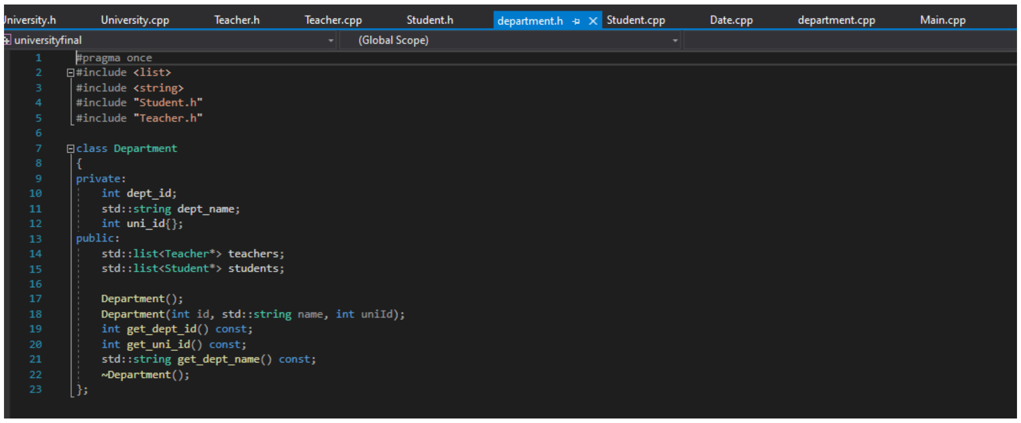

universityfinal
University
printUniversityDetails(const list<University*>& universities)
Explorer
Toolbox

```cpp
string University::get_university_name() const
{
    return this->university_name;
}

void University::printDepartmentDetails(University* university)
{
    cout << "****************************************" << endl;
    cout << "University Name: " << university->get_university_name() << endl;
    cout << "Total Departments: " << university->departments.size() << endl;
    if (!university->departments.empty())
        cout << "Sr. #\t\tDepartment name" << endl;
    for (auto* dept : university->departments)
    {
        cout << dept->get_dept_id() << "\t\t  " << dept->get_dept_name() << std::endl;

        if (!dept->teachers.empty())
        {
            cout << "Teachers in " << dept->get_dept_name() << " Department" << endl;
            cout << "Sr. #\tFirst Name \t Last Name \t Specialization \t Date of Employment" << endl;
            for (auto* teacher : dept->teachers)
            {
                cout << teacher->get_id() << "\t" << teacher->get_first_name() << "\t\t" << teacher->get_last_name() << "\t\t" << teacher->get_special
            }
        }

        if (!dept->students.empty())
        {
            cout << endl << endl;
            cout << "Students in " << dept->get_dept_name() << " Department" << endl;
```



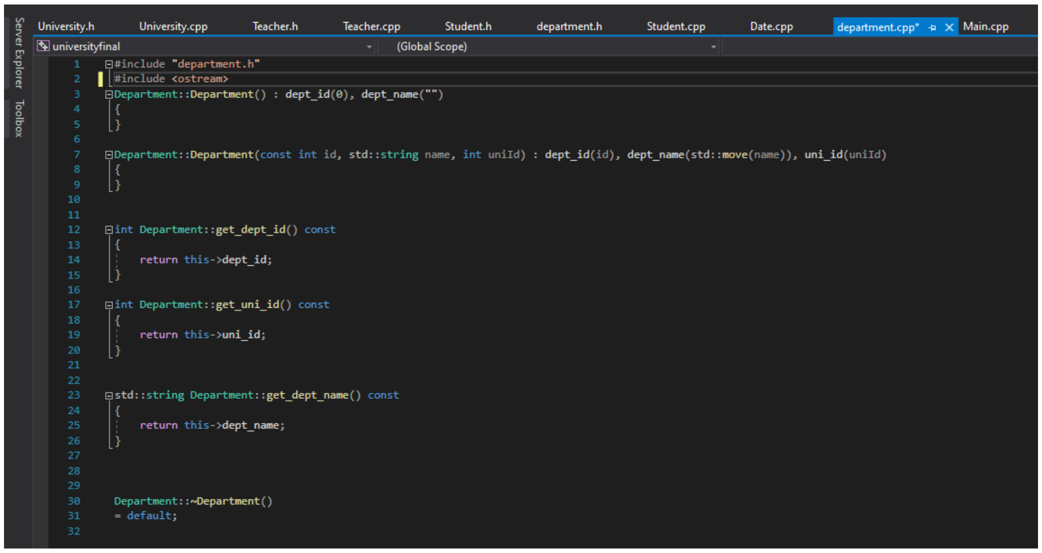
```cpp
                cout << "Sr. # \t First Name \t Last Name \t Grade" << endl;
                for (auto* student: dept->students)
                {
                    cout << student->get_id() << "\t\t" << student->get_first_name() << "\t\t" << student->get_last_name() << "\t\t" << student->get_grade
                }
            }

            cout << "****************************************" << endl;
            cout << endl;
        }
}

University::~University()
= default;
```

Department.h



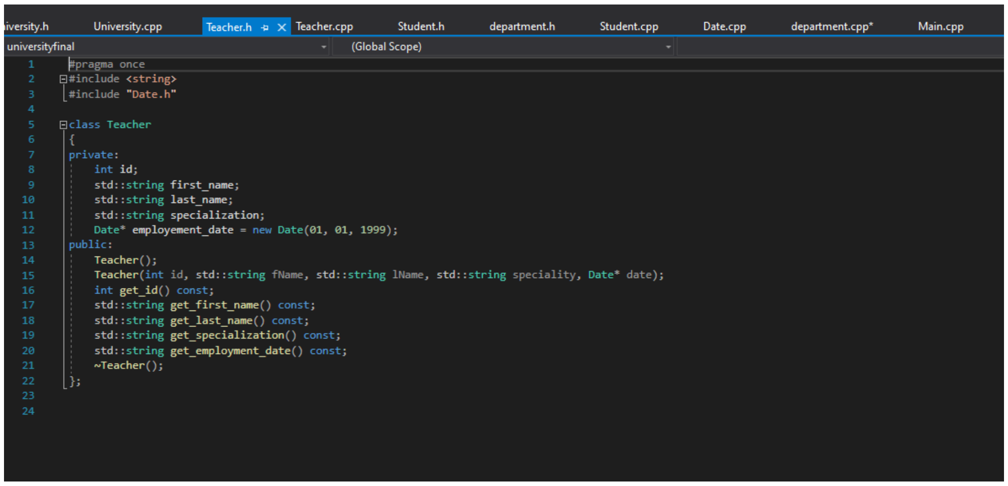
```cpp
#pragma once
#include <list>
#include <string>
#include "Student.h"
#include "Teacher.h"

class Department
{
private:
    int dept_id;
    std::string dept_name;
    int uni_id{};
public:
    std::list<Teacher*> teachers;
    std::list<Student*> students;

    Department();
    Department(int id, std::string name, int uniId);
    int get_dept_id() const;
    int get_uni_id() const;
    std::string get_dept_name() const;
    ~Department();
};
```

Department.cpp



```cpp
#include "department.h"
#include <ostream>
Department::Department() : dept_id(0), dept_name("")
{
}

Department::Department(const int id, std::string name, int uniId) : dept_id(id), dept_name(std::move(name)), uni_id(uniId)
{
}


int Department::get_dept_id() const
{
    return this->dept_id;
}

int Department::get_uni_id() const
{
    return this->uni_id;
}


std::string Department::get_dept_name() const
{
    return this->dept_name;
}



Department::~Department()
= default;

```

Teacher.h

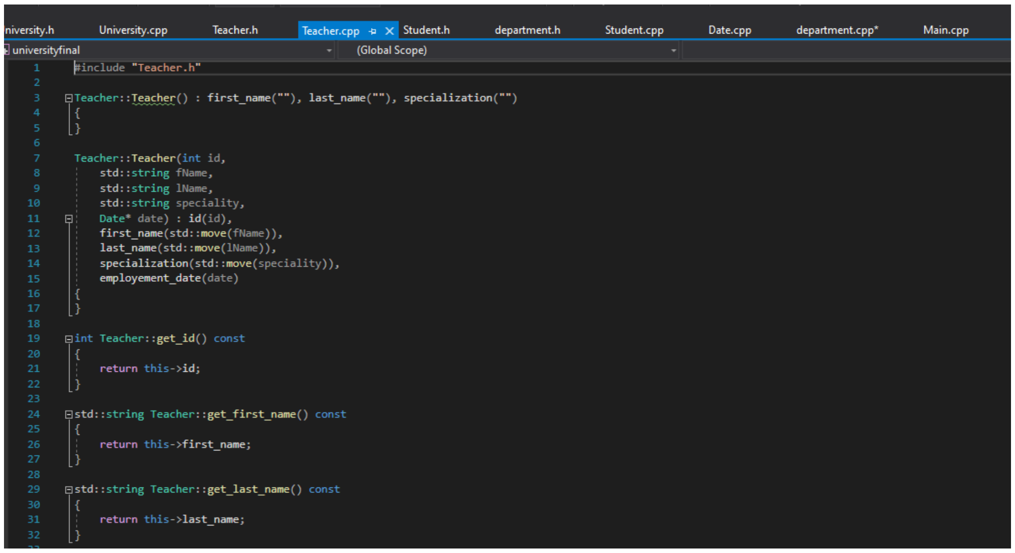


iversity.h | University.cpp | Teacher.h | Teacher.cpp | Student.h | department.h | Student.cpp | Date.cpp | department.cpp* | Main.cpp

universityfinal | (Global Scope)

```cpp
#pragma once
#include <string>
#include "Date.h"

class Teacher
{
private:
    int id;
    std::string first_name;
    std::string last_name;
    std::string specialization;
    Date* employement_date = new Date(01, 01, 1999);
public:
    Teacher();
    Teacher(int id, std::string fName, std::string lName, std::string speciality, Date* date);
    int get_id() const;
    std::string get_first_name() const;
    std::string get_last_name() const;
    std::string get_specialization() const;
    std::string get_employment_date() const;
    ~Teacher();
};


```

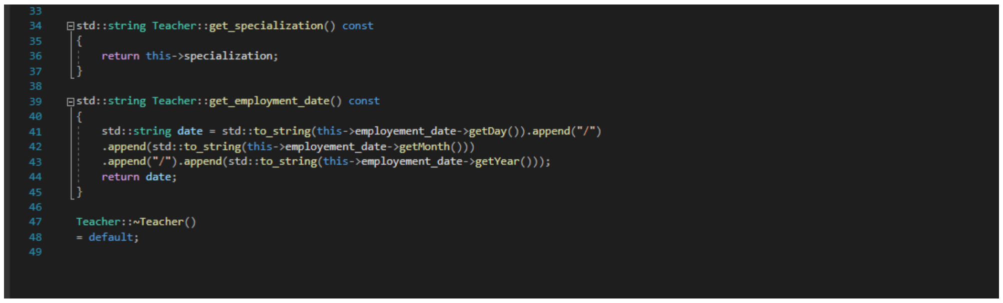
Teacher.cpp



niversity.h | University.cpp | Teacher.h | Teacher.cpp | Student.h | department.h | Student.cpp | Date.cpp | department.cpp* | Main.cpp

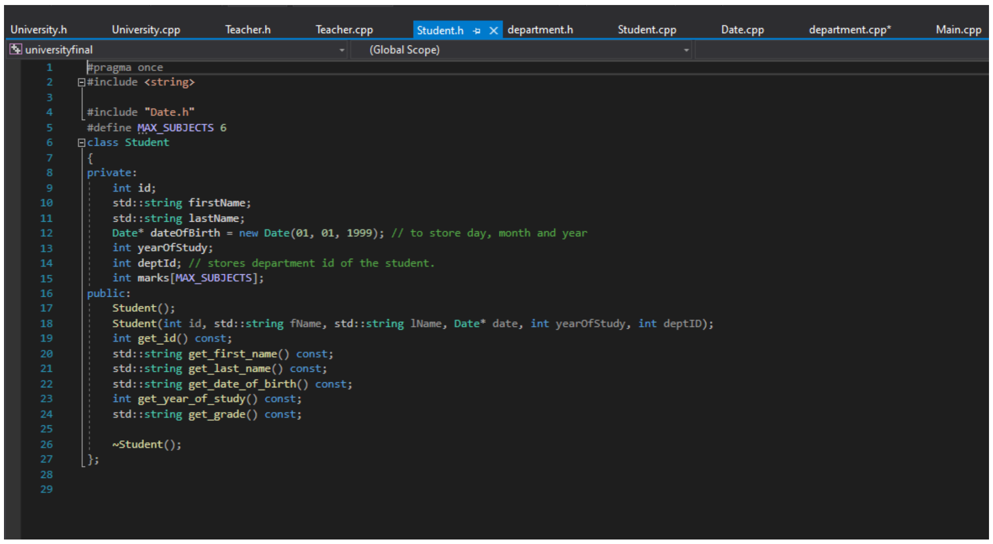
universityfinal | (Global Scope)

```cpp
#include "Teacher.h"

Teacher::Teacher() : first_name(""), last_name(""), specialization("")
{
}

Teacher::Teacher(int id,
    std::string fName,
    std::string lName,
    std::string speciality,
    Date* date) : id(id),
    first_name(std::move(fName)),
    last_name(std::move(lName)),
    specialization(std::move(speciality)),
    employement_date(date)
{
}

int Teacher::get_id() const
{
    return this->id;
}

std::string Teacher::get_first_name() const
{
    return this->first_name;
}

std::string Teacher::get_last_name() const
{
    return this->last_name;
}
```



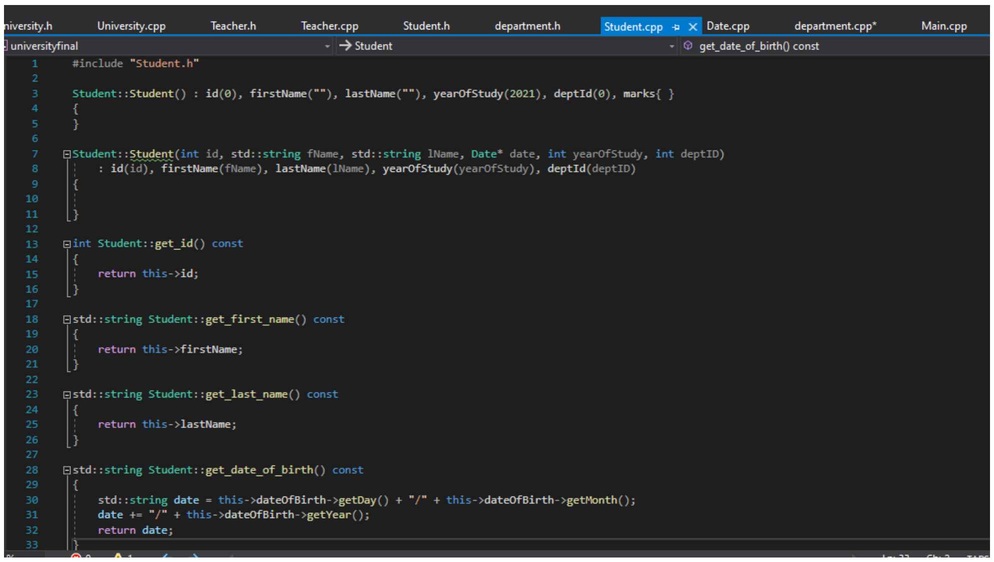
```cpp

std::string Teacher::get_specialization() const
{
    return this->specialization;
}

std::string Teacher::get_employment_date() const
{
    std::string date = std::to_string(this->employement_date->getDay()).append("/")
    .append(std::to_string(this->employement_date->getMonth()))
    .append("/").append(std::to_string(this->employement_date->getYear()));
    return date;
}

Teacher::~Teacher()
= default;

```

Student.h

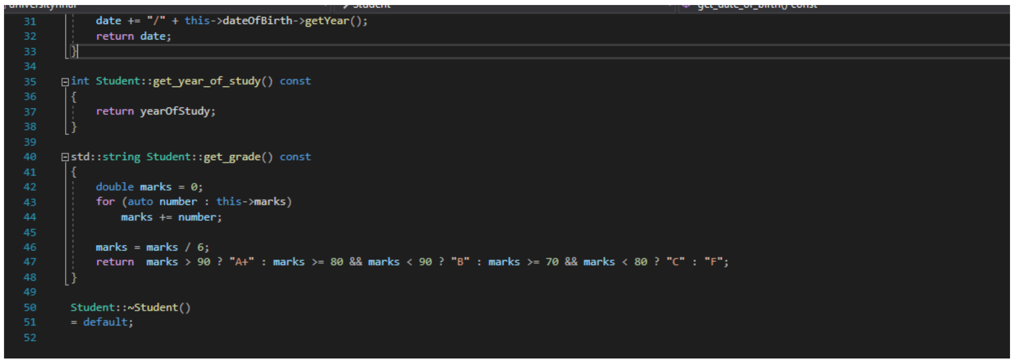

University.h University.cpp Teacher.h Teacher.cpp Student.h department.h Student.cpp Date.cpp department.cpp* Main.cpp
universityfinal
(Global Scope)

```cpp
#pragma once
#include <string>

#include "Date.h"
#define MAX_SUBJECTS 6
class Student
{
private:
    int id;
    std::string firstName;
    std::string lastName;
    Date* dateOfBirth = new Date(01, 01, 1999); // to store day, month and year
    int yearOfStudy;
    int deptId; // stores department id of the student.
    int marks[MAX_SUBJECTS];
public:
    Student();
    Student(int id, std::string fName, std::string lName, Date* date, int yearOfStudy, int deptID);
    int get_id() const;
    std::string get_first_name() const;
    std::string get_last_name() const;
    std::string get_date_of_birth() const;
    int get_year_of_study() const;
    std::string get_grade() const;

    ~Student();
};

```

Student.cpp


niversity.h University.cpp Teacher.h Teacher.cpp Student.h department.h Student.cpp Date.cpp department.cpp* Main.cpp
universityfinal
Student
get_date_of_birth() const

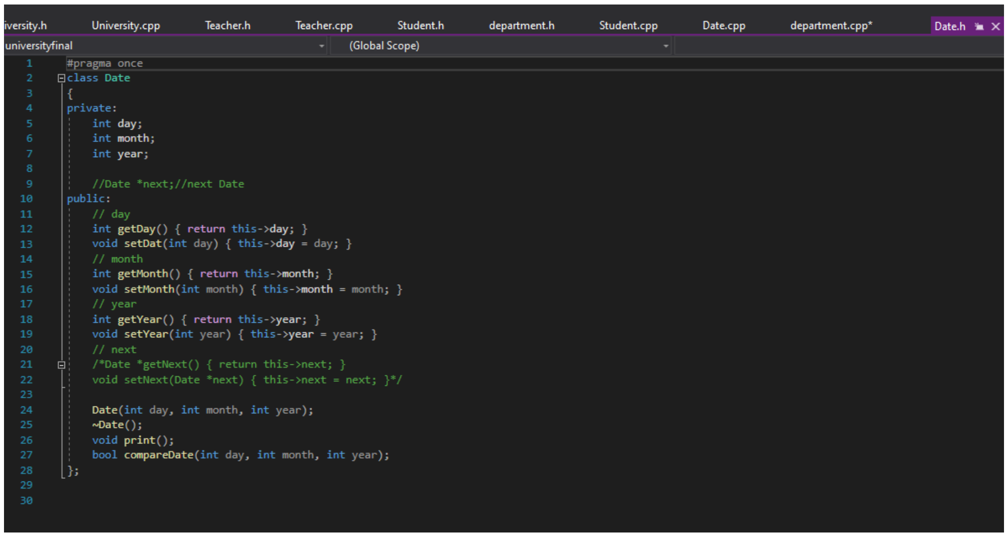
```cpp
#include "Student.h"

Student::Student() : id(0), firstName(""), lastName(""), yearOfStudy(2021), deptId(0), marks{ }
{
}

Student::Student(int id, std::string fName, std::string lName, Date* date, int yearOfStudy, int deptID)
    : id(id), firstName(fName), lastName(lName), yearOfStudy(yearOfStudy), deptId(deptID)
{

}

int Student::get_id() const
{
    return this->id;
}

std::string Student::get_first_name() const
{
    return this->firstName;
}

std::string Student::get_last_name() const
{
    return this->lastName;
}

std::string Student::get_date_of_birth() const
{
    std::string date = this->dateOfBirth->getDay() + "/" + this->dateOfBirth->getMonth();
    date += "/" + this->dateOfBirth->getYear();
    return date;
}
```



```cpp
    date += "/" + this->dateOfBirth->getYear();
    return date;
}
```

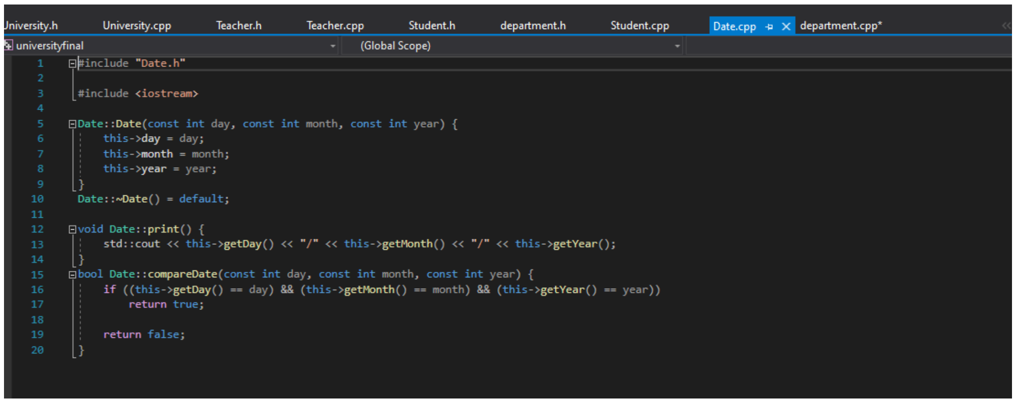
```cpp

int Student::get_year_of_study() const
{
    return yearOfStudy;
}

std::string Student::get_grade() const
{
    double marks = 0;
    for (auto number : this->marks)
        marks += number;

    marks = marks / 6;
    return  marks > 90 ? "A+" : marks >= 80 && marks < 90 ? "B" : marks >= 70 && marks < 80 ? "C" : "F";
}

Student::~Student()
= default;

```

Date.h



```cpp
#pragma once
class Date
{
private:
    int day;
    int month;
    int year;

    //Date *next;//next Date
public:
    // day
    int getDay() { return this->day; }
    void setDat(int day) { this->day = day; }
    // month
    int getMonth() { return this->month; }
    void setMonth(int month) { this->month = month; }
    // year
    int getYear() { return this->year; }
    void setYear(int year) { this->year = year; }
    // next
    /*Date *getNext() { return this->next; }
    void setNext(Date *next) { this->next = next; }*/

    Date(int day, int month, int year);
    ~Date();
    void print();
    bool compareDate(int day, int month, int year);
};


```

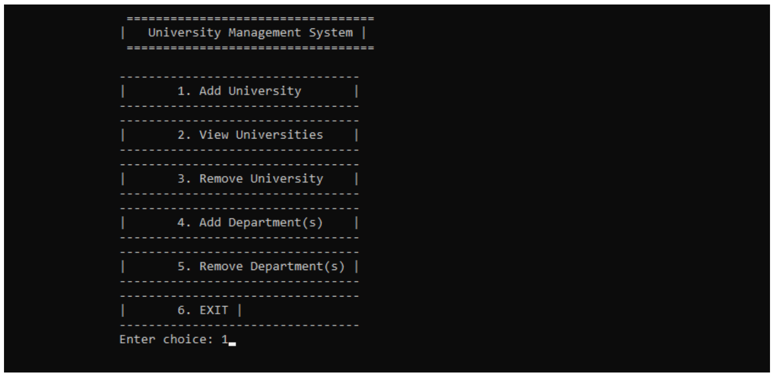
Date.cpp



```cpp
#include "Date.h"

#include <iostream>

Date::Date(const int day, const int month, const int year) {
    this->day = day;
    this->month = month;
    this->year = year;
}
Date::~Date() = default;

void Date::print() {
    std::cout << this->getDay() << "/" << this->getMonth() << "/" << this->getYear();
}
bool Date::compareDate(const int day, const int month, const int year) {
    if ((this->getDay() == day) && (this->getMonth() == month) && (this->getYear() == year))
        return true;

    return false;
}
```

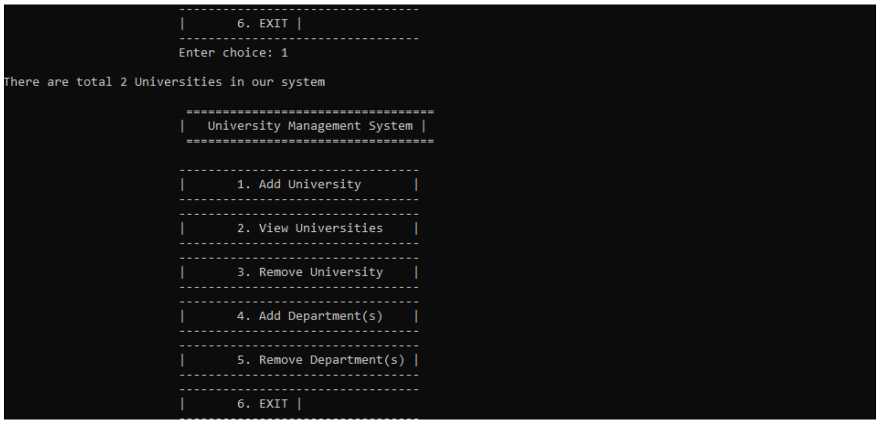
Testing:

Add university



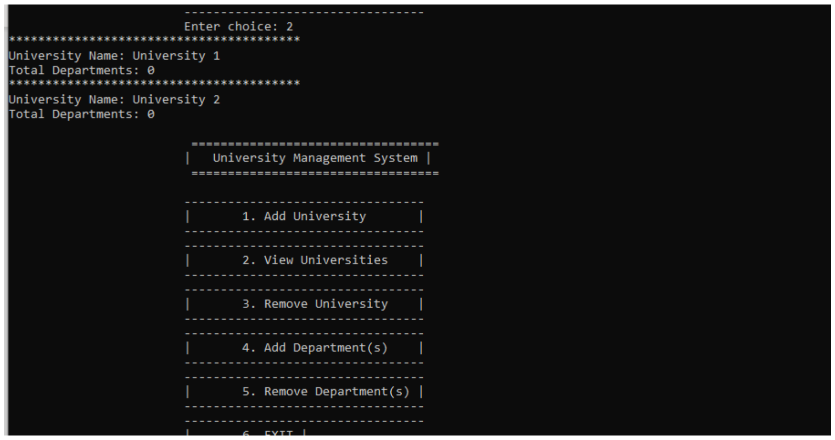
```
                 =================================
                |   University Management System |
                 =================================

                ---------------------------------
                |       1. Add University       |
                ---------------------------------
                ---------------------------------
                |       2. View Universities    |
                ---------------------------------
                ---------------------------------
                |       3. Remove University    |
                ---------------------------------
                ---------------------------------
                |       4. Add Department(s)    |
                ---------------------------------
                ---------------------------------
                |       5. Remove Department(s) |
                ---------------------------------
                ---------------------------------
                |       6. EXIT |
                ---------------------------------
                Enter choice: 1
```

Adding another university



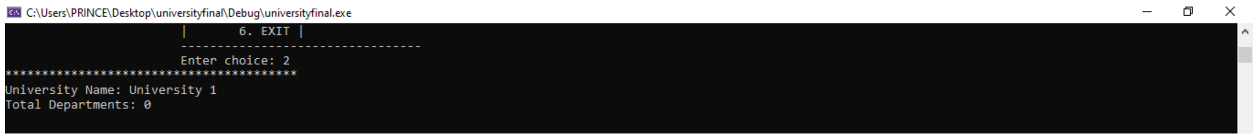
```
                        ---------------------------------
                        |       6. EXIT |
                        ---------------------------------
                        Enter choice: 1

There are total 2 Universities in our system

                         =================================
                        |   University Management System |
                         =================================

                        ---------------------------------
                        |       1. Add University       |
                        ---------------------------------
                        ---------------------------------
                        |       2. View Universities    |
                        ---------------------------------
                        ---------------------------------
                        |       3. Remove University    |
                        ---------------------------------
                        ---------------------------------
                        |       4. Add Department(s)    |
                        ---------------------------------
                        ---------------------------------
                        |       5. Remove Department(s) |
                        ---------------------------------
                        ---------------------------------
                        |       6. EXIT |
```

View university



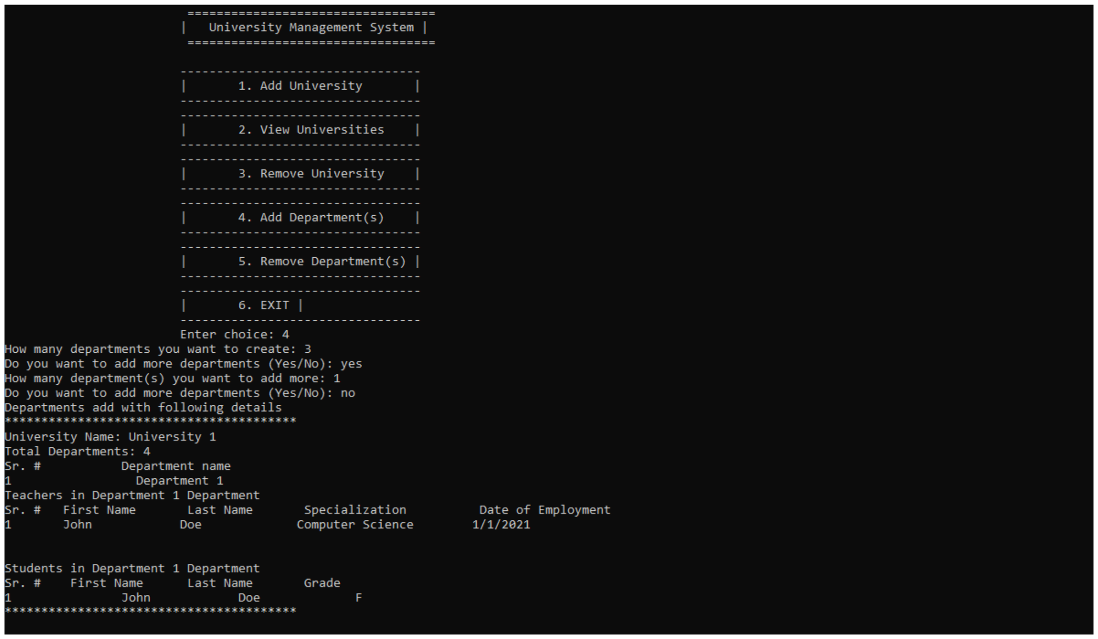
```
                        ---------------------------------
                        Enter choice: 2
****************************************
University Name: University 1
Total Departments: 0
****************************************
University Name: University 2
Total Departments: 0

                         =================================
                        |   University Management System |
                         =================================

                        ---------------------------------
                        |       1. Add University       |
                        ---------------------------------
                        ---------------------------------
                        |       2. View Universities    |
                        ---------------------------------
                        ---------------------------------
                        |       3. Remove University    |
                        ---------------------------------
                        ---------------------------------
                        |       4. Add Department(s)    |
                        ---------------------------------
                        ---------------------------------
                        |       5. Remove Department(s) |
                        ---------------------------------
                        ---------------------------------
                        |       6. EXIT |
```

Removing university

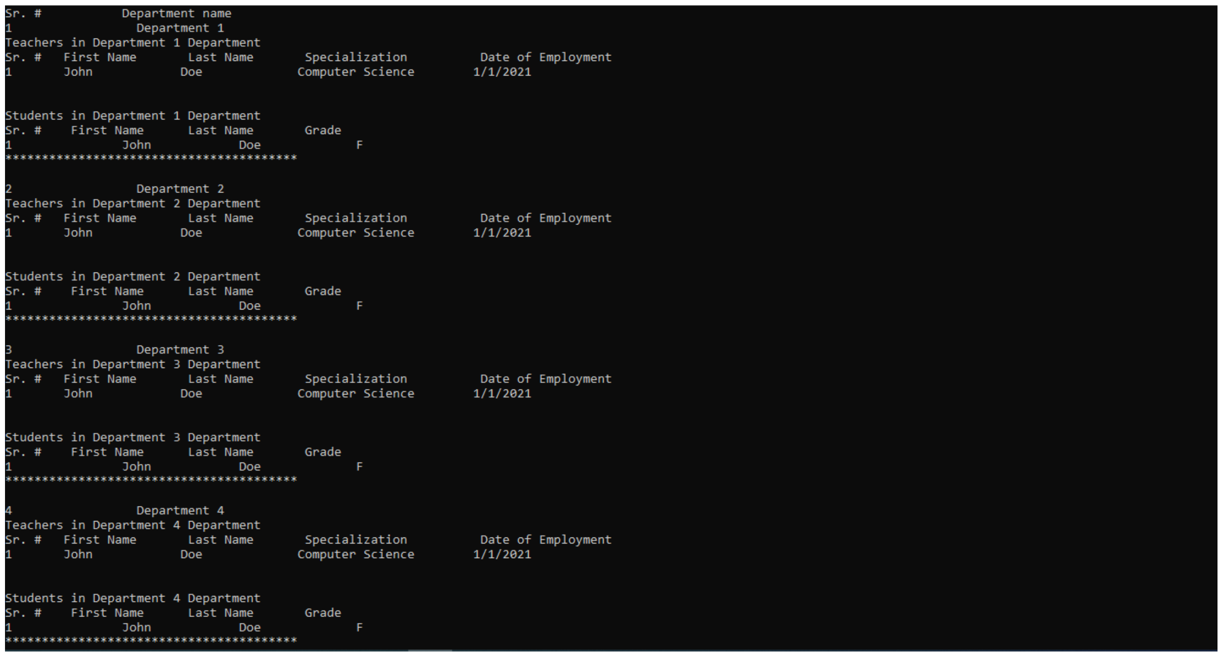


```
C:\Users\PRINCE\Desktop\universityfinal\Debug\universityfinal.exe
                        |       6. EXIT |
                        ---------------------------------
                        Enter choice: 2
****************************************
University Name: University 1
Total Departments: 0
```

Adding university with adding department and print information



```
                         ==================================
                        |   University Management System |
                         ==================================

                        ---------------------------------
                        |       1. Add University       |
                        ---------------------------------
                        ---------------------------------
                        |       2. View Universities    |
                        ---------------------------------
                        ---------------------------------
                        |       3. Remove University    |
                        ---------------------------------
                        ---------------------------------
                        |       4. Add Department(s)    |
                        ---------------------------------
                        ---------------------------------
                        |       5. Remove Department(s) |
                        ---------------------------------
                        ---------------------------------
                        |       6. EXIT |
                        ---------------------------------
                        Enter choice: 4
How many departments you want to create: 3
Do you want to add more departments (Yes/No): yes
How many department(s) you want to add more: 1
Do you want to add more departments (Yes/No): no
Departments add with following details
****************************************
University Name: University 1
Total Departments: 4
Sr. #           Department name
1                 Department 1
Teachers in Department 1 Department
Sr. #   First Name       Last Name       Specialization          Date of Employment
1       John            Doe             Computer Science        1/1/2021


Students in Department 1 Department
Sr. #    First Name      Last Name       Grade
1               John            Doe             F
****************************************
```



```
Sr. #           Department name
1                 Department 1
Teachers in Department 1 Department
Sr. #   First Name       Last Name       Specialization          Date of Employment
1       John            Doe             Computer Science        1/1/2021


Students in Department 1 Department
Sr. #    First Name      Last Name       Grade
1               John            Doe             F
****************************************

2                 Department 2
Teachers in Department 2 Department
Sr. #   First Name       Last Name       Specialization          Date of Employment
1       John            Doe             Computer Science        1/1/2021


Students in Department 2 Department
Sr. #    First Name      Last Name       Grade
1               John            Doe             F
****************************************

3                 Department 3
Teachers in Department 3 Department
Sr. #   First Name       Last Name       Specialization          Date of Employment
1       John            Doe             Computer Science        1/1/2021


Students in Department 3 Department
Sr. #    First Name      Last Name       Grade
1               John            Doe             F
****************************************

4                 Department 4
Teachers in Department 4 Department
Sr. #   First Name       Last Name       Specialization          Date of Employment
1       John            Doe             Computer Science        1/1/2021


Students in Department 4 Department
Sr. #    First Name      Last Name       Grade
1               John            Doe             F
****************************************
```