Student name :	Carter Hawks
Student email :	ckh170000@utdallas.edu
Class name :	2336.001_F18
Submitted on :	Oct 05, 2018 02:24 pm

Driver.cpp

```
#include <iostream>
#include <string>
using namespace std;
#include "Solution.cpp"

int main() {
        Person *p;
        int which;
        string name;
        int status = 0;
        int salary = 0;
        int rank = 0;
        string title = "0";

        cin >> which;
        cin >> name;
        switch (which) {
                case 1 :
                        p = new Person(name);
                        break;
                case 2 :
                        cin >> status;
                        p = new Student(name, status);
                        break;
                case 3 :
                        cin >> salary;
                        p = new Employee(name, salary);
                        break;
                case 4 :
                        cin >> salary;
                        cin >> rank;
                        p = new Faculty(name, salary, rank);
                        break;
                case 5 :
                        cin >> salary;
                        cin >> title;
                        p = new Staff(name, salary, title);
                        break;
        }
        cout << p->toString() << endl;    // dynamic binding of toString() same as (*p).toString();
        delete p;

}
```

Solution.cpp

```
/*

Analysis
    Given a person class, we have to make 4 derived classes from
    this one parent class.
Design
    Student and Employee both depend on Person, and we must reimplement their
    methods getClass and toString, as well as add getter and setters for
    their unique properties.
    Staff and Faculty both depend on Employee, and we must also implement
    or reimplement methods for those classes.

*/

#include <iostream>
#include <string>
using namespace std;

class Person {
        private:
                string name;
        public:
                Person(string name) {
                        this->name = name;
                }
                virtual ~Person() {
                }
                virtual string getClass() {
                        return "Person";
                }
                virtual string toString() {
            // Person name is David
                        return this->getClass() + " name is " + this->name;
                }
};

//Write your code here. Please Do Not change the Person class.

class Student: public Person {
    private:
        int classStatus;
    public:
        Student(string name, int classStatus):Person(name){
            this->classStatus = classStatus;
        }
        virtual ~Student() {
                }
                virtual string getClass() {
                        return "Student";
                }
        virtual string toString(){
            string className;
            switch(this->classStatus){
                case 1:
                    className = "FRESHMAN";
                    break;
                case 2:
                    className = "SOPHOMORE";
                    break;
                case 3:
                    className = "JUNIOR";
                    break;
                case 4:
                    className = "SENIOR";
                    break;
```

```
            }

            // Student name is Danial and status is JUNIOR
            return Person::toString() + " and status is " + className;
        }
};

class Employee: public Person {
    private:
        int salary;
    public:
        Employee(string name, int salary): Person(name) {
            this->salary = salary;
        }
        virtual ~Employee() {
                }
                virtual string getClass() {
                        return "Employee";
                }
        virtual string toString(){
            // Employee name is Emma and salary is 200
            return Person::toString() + " and salary is " + std::to_string(this->salary);
        }
};

class Faculty: public Employee {
    private:
        int rank;
    public:
        Faculty(string name, int salary, int rank): Employee(name, salary){
            this->rank = rank;
        }
        virtual ~Faculty(){
        }
        virtual string getClass(){
            return "Faculty";
        }
        virtual string toString(){
            // Faculty name is Emily and salary is 100 and rank is 5
            return Employee::toString() + " and rank is " + std::to_string(this->rank);
        }
};

class Staff: public Employee {
    private:
        string title;
    public:
        Staff(string name, int salary, string title): Employee(name, salary){
            this->title = title;
        }
        virtual ~Staff(){
        }
        virtual string getClass(){
            return "Staff";
        }
        virtual string toString(){
            // Staff name is Ava and salary is 900 and title is Manager
            return Employee::toString() + " and title is " + this->title;
        }
};
```

Name
Custom test case

Input
2 Danial 3

Output (Lines:2)
Student name is Danial and status is JUNIOR

Expected Output (Lines:0)

Status
NA

Name
Custom test case

Input
3 Emma 200

Output (Lines:2)
Employee name is Emma and salary is 200

Expected Output (Lines:0)

Status
NA

Name
Custom test case

Input
4 Emily 100 5

Output (Lines:2)
Faculty name is Emily and salary is 100 and rank is 5

Expected Output (Lines:0)

Status
NA

Name
Custom test case

Input
5 Ava 900 Manager

Output (Lines:2)

Staff name is Ava and salary is 900 and title is Manager

Expected Output (Lines:0)

Status
NA

Name
Default

Input
1 Mason

Output (Lines:2)
Person name is Mason

Expected Output (Lines:1)
Person name is Mason

Status
Pass
