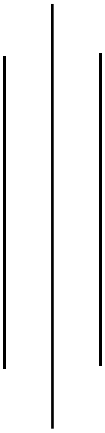
TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS



LAB REPORT ON

OBJECT ORIENTED PROGRAMMING [CT 451]

LAB 6

INHERITANCE IN C++

Submitted by:

Rujal Acharya

PUL076BEI029

Submitted to:

Department of Electronics and Computer Engineering, Pulchowk Campus

Institute of Engineering, Tribhuvan University

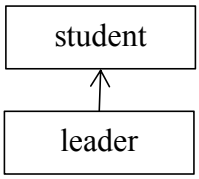
Lalitpur, Nepal

December, 2020

Task 1

Problem:

WAP in CPP to illustrate the concept of single inheritance using following diagram.



Program:

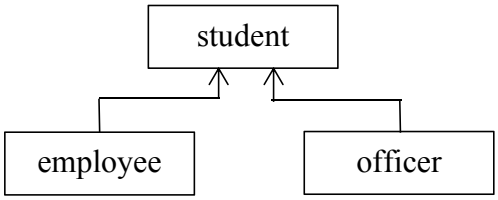
```
#include <iostream>

class Student {
	public:
		void getStudentData ( );

	protected:
		std::string name;
		std::string address;
		int roll;
};

class Leader : public Student {
	public:
		void getData ( );
		void showData ( );

	private:
		std::string unionName;
		int unionID;
};

void Student::getStudentData ( ) {
	std::cout << "Enter student name: ";
	std::cin >> this->name;
	std::cout << "Enter student address: ";
	std::cin >> this->address;
	std::cout << "Enter student roll no: ";
	std::cin >> this->roll;
}

void Leader::getData ( ) {
	this->getStudentData ( );
	std::cout << "Enter the name of union: ";
	std::cin >> this->unionName;
	std::cout << "Enter Union ID no: ";
	std::cin >> this->unionID;
```

```
}

void Leader::showData ( ) {
	std::cout << "Name: " << this->name << std::endl
			<< "Address: " << this->address << std::endl
			<< "Roll no: " << this->roll << std::endl
			<< "Name of union: " << this->unionName << std::endl
			<< "Union ID: " << this->unionID << std::endl;
}

int main ( ) {
	Leader leader;
	leader.getData ( );
	leader.showData ( );
	return EXIT_SUCCESS;
}
```

Task 2

Problem:

WAP in CPP to illustrate the concept of multiple inheritance using following diagram.



Program:

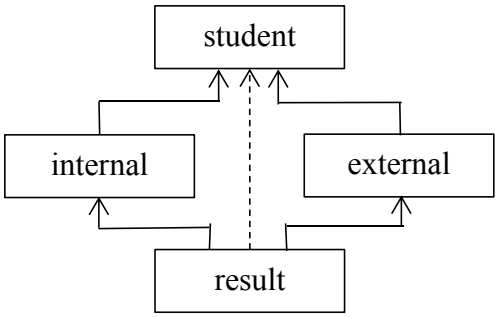
```
#include <iostream>

class Student {
	public:
		void getStudentData ( );

	protected:
		std::string university;
		std::string degree;
		int roll;
};

class Employee {
	public:
		void getEmployeeData ( );

	protected:
		std::string company;
		std::string position;
		int id;
};

class Officer: public Student, public Employee {
	public:
		void getData ( );
		void showData ( );

	private:
		std::string name;
};

void Student::getStudentData ( ) {
	std::cout << "Enter the name of university: ";
	std::cin >> this->university;
	std::cout << "Enter the degree: ";
	std::cin >> this->degree;
```

```
    std::cout << "Enter roll no: ";
    std::cin >> this->roll;
}

void Employee::getEmployeeData ( ) {
    std::cout << "Enter the name of company: ";
    std::cin >> this->company;
    std::cout << "Enter the employee position: ";
    std::cin >> this->position;
    std::cout << "Enter the employee id: ";
    std::cin >> this->id;
}

void Officer::getData ( ) {
    std::cout << "Enter the name of officer: ";
    std::cin >> this->name;
    this->getStudentData ( );
    this->getEmployeeData ( );
}

void Officer::showData ( ) {
    std::cout << "Name: " << this->name << std::endl
              << "University: " << this->university << std::endl
              << "Degree: " << this->degree << std::endl
              << "Roll No: " << this->roll << std::endl
              << "Company: " << this->company << std::endl
              << "Position: " << this->position << std::endl
              << "Employee ID: " << this->id << std::endl;
}

int main ( ) {
    Officer officer;
    officer.getData ( );
    officer.showData ( );
    return EXIT_SUCCESS;
}
```

Task 3

Problem:

WAP in CPP to illustrate the concept of multipath inheritance and virtual base class using following diagram.



Program:

```
#include <iostream>

class Student {
	public:
		void getStudentData ( );

	protected:
		std::string name;
		std::string address;
		int roll;
};

class Internal : virtual public Student {
	public:
		void getInternalData ( );

	protected:
		float internalPercentage;
};

class External : virtual public Student {
	public:
		void getExternalData ( );

	protected:
		float externalPercentage;
};

class Result : public Internal, public External {
	public:
		void getData ( );
		void showData ( );
```

```
    private:
        float totalPercentage;
};

void Student::getStudentData ( ) {
    std::cout << "Enter the student name: ";
    std::cin >> this->name;
    std::cout << "Enter the address: ";
    std::cin >> this->address;
    std::cout << "Enter the roll number: ";
    std::cin >> this->roll;
}

void Internal::getInternalData ( ) {
    std::cout << "Enter percentage in internal exam: ";
    std::cin >> this->internalPercentage;
}

void External::getExternalData ( ) {
    std::cout << "Enter percentage in external exam: ";
    std::cin >> this->externalPercentage;
}

void Result::getData ( ) {
    this->getStudentData ( );
    this->getInternalData ( );
    this->getExternalData ( );
    this->totalPercentage = 0.2 * this->internalPercentage + 0.8 * this->externalPercentage;
}

void Result::showData ( ) {
    std::cout << "Name: " << this->name << std::endl
              << "Address: " << this->address << std::endl
              << "Roll No: " << this->roll << std::endl
              << "Internal Percentage: " << this->internalPercentage << std::endl
              << "External Percentage: " << this->externalPercentage << std::endl
              << "Final Percentage: " << this->totalPercentage << std::endl;
}

int main ( ) {
    Result result;
    result.getData ( );
    result.showData ( );
    return EXIT_SUCCESS;
}
```

Task 4

Problem:

WAP in CPP to illustrate the concept of constructor and destructor invocation in single inheritance.

Program:

```
#include <iostream>

class Parent {
	public:
		Parent ( );
		~ Parent ( );
};

class Child : public Parent {
	public:
		Child ( );
		~ Child ( );
};

Parent::Parent ( ) {
	std::cout << "Inside constructor of parent class..." << std::endl;
}

Parent :: ~ Parent ( ) {
	std::cout << "Inside destructor of parent class..." << std::endl;
}

Child::Child ( ) {
	std::cout << "Inside constructor of child class..." << std::endl;
}

Child :: ~ Child ( ) {
	std::cout << "Inside destructor of child class..." << std::endl;
}

int main ( ) {
	Child child;
	return EXIT_SUCCESS;
}
```

Task 5

Problem:

WAP in CPP to illustrate the concept of constructor and destructor invocation in multiple inheritance.

Program:

```
#include <iostream>

class High {
	public:
		High ( );
		~ High ( );
};

class Middle : public High {
	public:
		Middle ( );
		~ Middle ( );
};

class Low : public Middle {
	public:
		Low ( );
		~ Low ( );
};

High::High ( ) {
	std::cout << "Inside constructor of class with highest hierarchy..." << std::endl;
}

High :: ~ High ( ) {
	std::cout << "Inside destructor of class with highest hierarchy..." << std::endl;
}

Middle::Middle ( ) {
	std::cout << "Inside constructor of class with middle hierarchy..." << std::endl;
}

Middle :: ~ Middle ( ) {
	std::cout << "Inside destructor of class with middle hierarchy..." << std::endl;
}

Low::Low ( ) {
	std::cout << "Inside constructor of class with lowest hierarchy..." << std::endl;
}

Low :: ~ Low ( ) {
	std::cout << "Inside destructor of class with lowest hierarchy..." << std::endl;
```

```
}

int main ( ) {
	Low low;
	return EXIT_SUCCESS;
}
```