CENG201 – OOP - Implementation Assignment 2

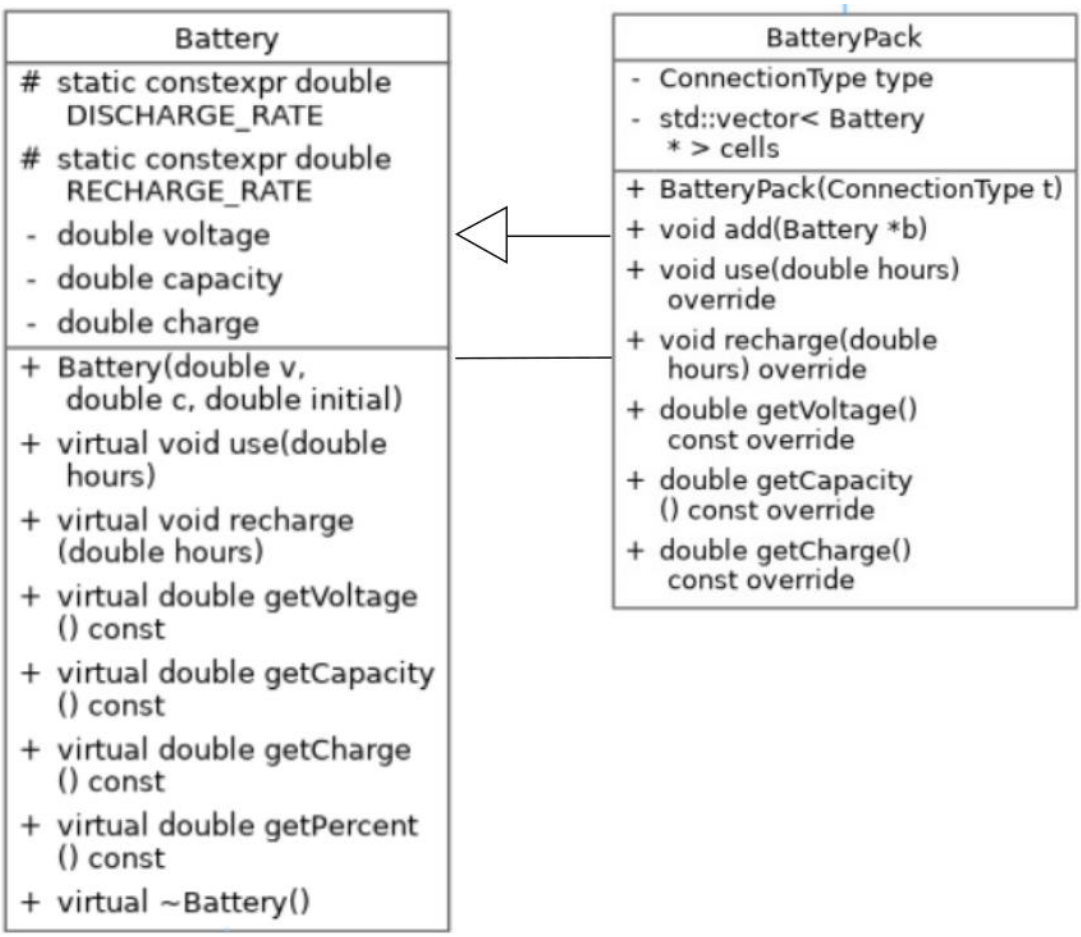
This is a group assignment, each project group will do the it together. In this assignment, you will implement two C++ classes:

1. Battery
2. BatteryPack (inherits from Battery)

This assignment focuses entirely on inheritance, composition, polymorphism, virtual methods, recursive structures, Doxygen documentation, proper header/source file separation

You will also write a small console test program.

Class Explanations



1. Implement Class: Battery

Attributes

- double voltage: voltage in volts
- double capacity: capacity in mAh
- double charge: currect charge in mAh
- double DISCHARGE_RATE: static constexpr variable fixed to 100.
- double RECHARGE_RATE: static constexpr variable fixed to 150.

Constructor

- Battery(double v, double c, double initialCharge);

Public Methods

- void use(double hours): reduces charge by hours * DISCHARGE_RATE. Cannot go below 0.
- void recharge(double hours); increases charge by hours * RECHARGE_RATE. Cannot go above capacity.
- double getVoltage() const;
- double getCapacity() const;
- double getCharge() const;
- double getPercent() const; gives the charge in percent relative to the capacity (=100*charge/capacity).
- virtual ~Battery() {}

Behavior:

- A fixed discharge rate (100 mAh/hour)
- A fixed recharge rate (150 mAh/hour)
- Charge is always clamped between [0, capacity]

---

2. Implement Class: BatteryPack

Inheritance

BatteryPack **is-a** Battery.

Attributes

- enum ConnectionType { SERIES, PARALLEL };
- ConnectionType type;
- std::vector<Battery*> cells;

Constructor

- BatteryPack(ConnectionType t);

Methods to Implement

- void add(Battery* b);

- void use(double hours) override;
- void recharge(double hours) override;
- double getVoltage() const override;
- double getCapacity() const override;
- double getCharge() const override;

Computation Rules

Series

Voltage = sum of all voltages

Capacity = minimum capacity among cells

Charge = minimum charge among cells

Parallel

Voltage = voltage of any cell

Capacity = sum of capacities

Charge = sum of charges

Discharging / Recharging

use() and recharge() must simply call each child's method: Not physically accurate but for simplicity.

```
for (Battery* b : cells)
   b->use(hours);
```

---

3. Console Test Program (BatteryTest.cpp)

You can create a test program to test your classes. For instance, create several Battery objects, add them into a BatteryPack. Check the methods and attributes.

---

4. Required Project Structure

Battery.h

Battery.cpp

BatteryPack.h

BatteryPack.cpp

BatteryTest.cpp

---

5. Required Documentation Format: DOXYGEN

Each class, method, and attribute must have a Doxygen comment block:

Example:

```
/**
 * @brief Decreases charge based on a fixed discharge rate.
 * @param hours Number of hours of usage.
 */
void use(double hours);
```

Points will be deducted for missing documentation.

---

6. Compilation Instructions (Ubuntu)

You can compile using g++:

g++ Battery.cpp BatteryPack.cpp BatteryTest.cpp -o battery_test

Run with:

./battery_test

---

7. Sample test

```
#include <iostream>
#include "Battery.h"
#include "BatteryPack.h"

int main() {

   Battery b1(3.7, 2000, 1500);
   Battery b2(3.7, 2000, 1800);
   Battery b3(3.7, 2000, 2000);

   BatteryPack pack(BatteryPack::SERIES);
   pack.add(&b1);
   pack.add(&b2);
   pack.add(&b3);

   std::cout << "Initial:\n";
   std::cout << "Voltage: " << pack.getVoltage() << "\n";
   std::cout << "Capacity: " << pack.getCapacity() << "\n";
   std::cout << "Charge: " << pack.getCharge() << "\n\n";

   pack.use(1.0);

   std::cout << "After 1 hour use:\n";
   std::cout << "Charge: " << pack.getCharge() << "\n\n";

   pack.recharge(1.0);

   std::cout << "After 1 hour recharge:\n";
   std::cout << "Charge: " << pack.getCharge() << "\n";

   return 0;
}
```

Output:

```
Initial:
Voltage: 11.1
Capacity: 2000
Charge: 1500
```

After 1 hour use:
Charge: 1400

After 1 hour recharge:
Charge: 1550

---

7. Submission

- Put all your source files into an archive with name Assignment2.zip and submit to aybuzem.
- Only one member from each group must submit the assignment.
- Also, put your source files into your group's github repo.

To Be Continued ... (In assignment 3)