Milestone 1

```cpp
// Opgave 1A
// VaerdiPapir.h
#pragma once
#include <string>
using namespace std;

class VaerdiPapir
{
public:
    VaerdiPapir(string name = "", double kurs = 100.0);
    // virtual destructor vigtig
    virtual ~VaerdiPapir();

    void setKurs(double);
    double getKurs() const;

    void setName(string);
    string getName() const;

    // pure virtual vigtigt
    virtual double beregnHandelsVaerdi(double) const = 0;

protected:
    double kurs_;
private:
    string name_;
};

// Vaerdipapir.cpp
#include "VaerdiPapir.h"

VaerdiPapir::VaerdiPapir(string name, double kurs)
{
    setKurs(kurs);
    setName(name);
}

VaerdiPapir::~VaerdiPapir()
{

}

void VaerdiPapir::setKurs(double kurs)
{
    kurs_ = (kurs >= 0 ? kurs : 100.0);
}

double VaerdiPapir::getKurs() const
{
    return kurs_;
}

void VaerdiPapir::setName(string name)
{
    name_ = name;
}
```

```cpp
string VaerdiPapir::getName() const
{
    return name_;
}

// Opgave 1B
// Aktie.h
#pragma once

#include "VaerdiPapir.h"

class Aktie : public VaerdiPapir
{
public:
    Aktie(string name = "", double kurs = 100, double stykStoerrelse = 100);


    void setStykstoerrrelse(double);
    double getStykStoerrelse() const;

    // override er option
    double beregnHandelsVaerdi(double antal) const override;

private:
    double stykStoerrelse_;
};

// Aktie.cpp
#include "Aktie.h"



Aktie::Aktie(string name, double kurs, double stykStoerrelse) : VaerdiPapir(name, kurs)
{
    setStykstoerrrelse(stykStoerrelse);
}

void Aktie::setStykstoerrrelse(double ss)
{
    stykStoerrelse_ = ss < 0 ? 100 : ss;
}

double Aktie::getStykStoerrelse() const
{
    return stykStoerrelse_;
}

double Aktie::beregnHandelsVaerdi(double antal) const
{
    return antal * kurs_;
}
```

Milestone 2.

```
// Opgave 1C
// Obligation.h
#pragma once
#include "VaerdiPapir.h"

class Obligation : public VaerdiPapir
{
public:
    Obligation(string name = "", double kurs = 100.0, double rente = 2.0);

    void setRenteSats(double rente);
    double getRenteSats() const;

    // Override er option
    double beregnHandelsVaerdi(double antal) const override;

private:
    double renteSats_;
};

// Obligation.cpp
#include "Obligation.h"

Obligation::Obligation(string name, double kurs, double rente) : VaerdiPapir(name, kurs)
{
    setRenteSats(rente);
}

void Obligation::setRenteSats(double rente)
{
    renteSats_ = (0.0 <= rente ? rente : 0.0);
}

double Obligation::getRenteSats() const
{
    return renteSats_;
}

double Obligation::beregnHandelsVaerdi(double antal) const
{
    return antal * kurs_ / 100;
}

// Opgave 1D

// Main program

#include "VaerdiPapir.h"
#include "Aktie.h"
#include "Obligation.h"

#include <iostream>
using namespace std;

int main()
{
    VaerdiPapir * arr[2];
```

```
    arr[0] = new Aktie("B&O", 101.5, 50.0);
    arr[1] = new Obligation("RD 16. serie 2047", 99.5, 2.0);

    for (int i = 0; i < 2; i++)
    {
        cout << arr[i]->beregnHandelsVaerdi(1000) << endl;
    }

    return 0;
}
```

```
101500
995
Press any key to continue . . .
```

```
// Opgave 2A+B

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

class BudgetPost
{
public:
    BudgetPost(string, string, double);

    void setTekst(string);
    string getTekst() const;

    void setKategori(string);
    string getKategori() const;

    void setBeloeb(double);
    double getBeloeb() const;

private:
    string tekst_;
    string kategori_;
    double beloeb_;
};

ostream & operator<<(ostream &, const BudgetPost &);

bool operator <(const BudgetPost &, const BudgetPost &);

// BudgetPost.cpp
#include "BudgetPost.h"

BudgetPost::BudgetPost(string tekst, string kategori, double belob)
{
    setTekst(tekst);
    setKategori(kategori);
    setBeloeb(belob);
}

void BudgetPost::setTekst(string tekst)
{
    tekst_ = tekst;
}

string BudgetPost::getTekst() const
{
    return tekst_;
}

void BudgetPost::setKategori(string kat)
{
    kategori_ = kat;
}

string BudgetPost::getKategori() const
{
```

```
    return kategori_;
}

void BudgetPost::setBeloeb(double b)
{
    beloeb_ = b;
}

double BudgetPost::getBeloeb() const
{
    return beloeb_;
}

ostream & operator<<(ostream & os, const BudgetPost & b)
{
    os << b.getTekst() << " (" << b.getKategori() << ") " << b.getBeloeb();

    return os;
}

bool operator<(const BudgetPost & l, const BudgetPost & r)
{
    return l.getKategori() < r.getKategori();
}

// Opgave 2c
#include <iostream>
#include <iomanip>

using namespace std;

#include "BudgetPost.h"

int main()
{
    BudgetPost t1("DSB", "Transport", -325.00);
    BudgetPost t2("Kvickly", "Dagligvarer", -987.67);
    BudgetPost t3("Cirkle K", "Transport", -476.98);
    BudgetPost t4("DSB", "Transport", -275.00);
    BudgetPost t5("Rema 1000", "Dagligvarer", -76.98);

    // Sæt 2 decimaler på alle følgende udskrifter
    cout << setprecision(2) << fixed;

    cout << "Posten t1: " << t1 << endl;
    cout << "Posten t2: " << t2 << endl;
    cout << "Posten t3: " << t3 << endl;
    cout << "Posten t4: " << t4 << endl;
    cout << "Posten t5: " << t5 << endl;

    cout << endl << endl;

    return 0;
}
```

```
Posten t1: DSB (Transport) -325.00
Posten t2: Kvickly (Dagligvarer) -987.67
Posten t3: Cirkle K (Transport) -476.98
Posten t4: DSB (Transport) -275.00
Posten t5: Rema 1000 (Dagligvarer) -76.98
Press any key to continue . . .
```

Milestone 4.

```
// Opgave 2D
// Budget.h
#pragma once
#include <vector>
using namespace std;
#include "BudgetPost.h"

class Budget
{
public:
    Budget& indsaetPost(const BudgetPost &);
    void print() const;
    void printKategori(string) const;

private:
    vector<BudgetPost> poster_;
};

// Budget.cpp
#include "Budget.h"
#include <iostream>
using namespace std;

Budget & Budget::indsaetPost(const BudgetPost &p)
{
    poster_.push_back(p);
    return *this;
}

void Budget::printKategori(string k) const
{
    double sum = 0;
    for (vector<BudgetPost>::const_iterator it = poster_.cbegin(); it != poster_.cend();
         it++)
    {
        if (it->getKategori() == k)
        {
            cout << *it << endl;
            sum += it->getBeloeb();
        }
    }

    cout << endl << "Ialt udgifter for " << sum << " for kategori " << k << endl;
}

// Opgave 2E

int main()
{
    // .....

    Budget budget;

    budget.indsaetPost(t1).indsaetPost(t2).indsaetPost(t3).indsaetPost(t4).indsaetPost(t5);
```

```
    budget.printKategori("Dagligvarer");

    // ............
}
```

```
Kvickly (Dagligvarer) -987.67
Rema 1000 (Dagligvarer) -76.98

Ialt udgifter for -1064.65 for kategori Dagligvarer
```

Milestone 5

```
// Budget.cpp
void Budget::print() const
{
    vector<BudgetPost> temp(poster_);

    sort(temp.begin(), temp.end());

    cout << "Budgettet har " << temp.size() << " poster" << endl << endl;

    for (vector<BudgetPost>::iterator it = temp.begin(); it != temp.end(); it++)
    {
        cout << *it << endl;
    }

}

// main
int main()
{
    // ...........

    budget.print();

    // ............
}
```

```
Budgettet har 5 poster

Kvickly (Dagligvarer) -987.67
Rema 1000 (Dagligvarer) -76.98
DSB (Transport) -325.00
Cirkle K (Transport) -476.98
DSB (Transport) -275.00
Press any key to continue . . .
```