OOP reeksamen F16 – løsningsforslag

OPGAVE 1

```
// Milestone 1 (StraightLine.cpp)
StraightLine::StraightLine(int slope, int yCross)
{
	slope_ = slope;
	yCross_ = yCross;
}

// Milestone 1 (main.cpp)
int main()
{
	StraightLine line1(2, 4);
	StraightLine line2(-3, 2);

	line1.print();
	line2.print();

	return 0;
}

// Milestone 2 (StraightLine.h)
istream & operator>>(istream& in, StraightLine & line);

bool StraightLine::operator==(const StraightLine & rightLine);

// Milestone 2 (StraightLine.cpp)
bool StraightLine::operator==(const StraightLine & rightLine)
{
	return (slope_ == rightLine.slope_ && yCross_ == rightLine.yCross_);
}

istream & operator>>(istream & in, StraightLine & line)
{
	int a, b;

	in >> a >> b;

	line.setSlope(a);
	line.setYCross(b);

	return in;
}


	// Milestone 2 (main.cpp)
	StraightLine line3;

	cin >> line3;

	if (line3 == line1 || line3 == line2)
		cout << "Den indtastede linje findes allerede" << endl;
	else
		cout << "Den indtastede linje findes ikke i forvejen" << endl;
	// Milestone 3 (main.cpp)
	vector<StraightLine> myVector;

	myVector.push_back(line1);
	myVector.push_back(line2);
	myVector.push_back(line3);

	vector<StraightLine>::iterator myIter;

	for (myIter = myVector.begin(); myIter != myVector.end(); ++myIter)
	{
		myIter->print();
	}
```

OPGAVE 2

```
// Milestone 4 (BinaryCache.cpp)
BinaryCache::BinaryCache(int capacity)
{
	capacity_ = (capacity > 1 ? capacity : 1);
	size_ = 0;
	numberPtr_ = new int[capacity_];
}

void BinaryCache::addNumber(int number)
{
	if (number == 0 || number == 1){
		if (capacity_ == size_){
			int *temp = numberPtr_;

			capacity_ *= 2;

			numberPtr_ = new int[capacity_];

			for (int i = 0; i < size_; i++)
				numberPtr_[i] = temp[i];

			delete[] temp;
		}
		numberPtr_[size_++] = number;
	}
}

// Milestone 4 (main.cpp)
int main()
{
	BinaryCache myCache(2);

	myCache.addNumber(1);
	myCache.addNumber(0);
	myCache.addNumber(1);

	cout << myCache.getSize() << endl;
	cout << myCache.getCapacity() << endl;
	myCache.print();

	return 0;
}
```

```cpp
// Milestone 5 (BinaryCache.cpp)
const BinaryCache & BinaryCache::operator=(const BinaryCache & right)
{
	if (this != &right)
	{
		if (capacity_ != right.capacity_)
		{
			delete [] numberPtr_;

			capacity_ = right.capacity_;
			size_ = right.size_;

			numberPtr_ = new int[capacity_];
		}

		for (int i = 0; i < size_; i++)
			numberPtr_[i] = right.numberPtr_[i];
	}
	return *this;
}

BinaryCache::~BinaryCache()
{
	delete[] numberPtr_;
}
```

OPGAVE 3

```cpp
// Milestone 6 (Insurance.h)
	virtual double calculatePrice() const = 0;
	virtual void print() const;
protected:
	string policynumber_;

// Milestone 6 (Insurance.cpp)
Insurance::Insurance(string policynumber, double standardPrice, int deductible)
{
	policynumber_ = policynumber;
	standardPrice_ = (standardPrice >= 500 ? standardPrice : 500);
	deductible_ = (0 <= deductible && deductible <= 5000 ? deductible : 5000);
}

// Milestone 6 (CarInsurance.h)
class CarInsurance : public Insurance
{
public:
	virtual double calculatePrice() const;
	virtual void print() const;

// Milestone 6 (CarInsurance.cpp)
CarInsurance::CarInsurance(string policynumber, double standardPrice, int
deductible, int kmPerYear)
: Insurance(policynumber, standardPrice, deductible)
{
	kmPerYear_ = (kmPerYear >= 10000 ? kmPerYear : 10000);
}
```

```cpp
// Milestone 6 (main.cpp)
int main()
{
     Insurance * myPtr = new CarInsurance("AB-10123456", 5000.00, 2500, 20000);

     myPtr->print();

     return 0;
}
```