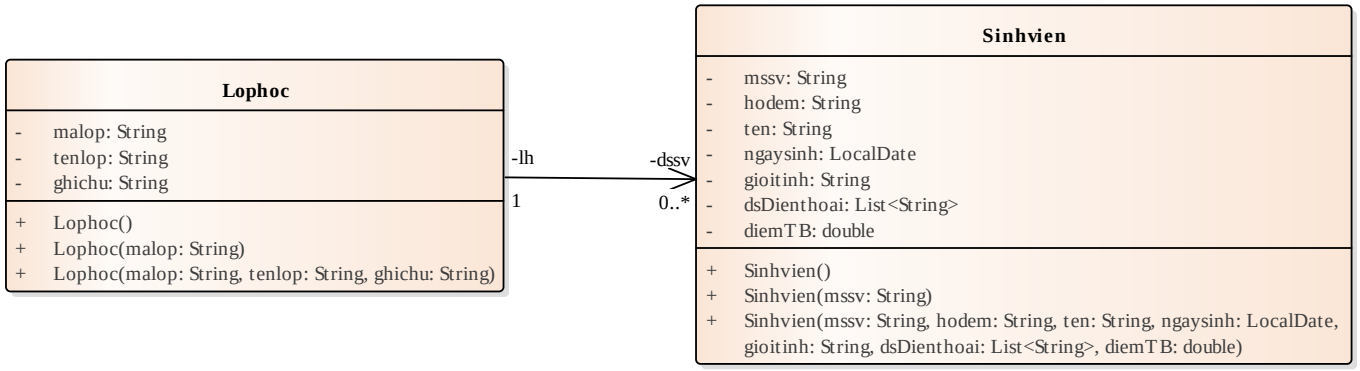
Bài tập tổng hợp

Với mô hình lớp sau



Dữ liệu mẫu


```
{
	"malop": "DHKHMT13A",
	"tenlop": "Đại học Khoa học Máy tính 13A",
	"ghichu": "Công nghệ thông tin",
	"dssv": [
		{
			"mssv": "17640461",
			"hodem": "Đặng Lê",
			"ten": " Nguyên",
			"ngaysinh": {
				"year": 1999,
				"month": 4,
				"day": 10
			},
			"gioitinh": "Nam",
			"dsDienthoai": [
				"0903603220",
				"0913612261"
			],
			"diemTB": 8.21
		},
		{
			"mssv": "17659941",
			"hodem": "Trần Trọng",
			"ten": "Bình",
			"ngaysinh": {
				"year": 1999,
				"month": 12,
				"day": 11
			},
```

```
                "gioitinh": "Nam",
                "dsDienthoai": [
                        "0903521617"
                ],
                "diemTB": 5.83
        }
    ]
}
```

Yêu cầu

1/ Tạo kết nối với MongoDB
2/ Viết các lớp thực thể của mô hình lớp
3/ Viết các lớp xử lý dữ liệu (CRUD)
4/ Viết thêm một số câu truy vấn thực thi các yêu cầu.

Phần hướng dẫn

Eclipse Class header

Eclipse → Window → Preferences → Java -> Code Style → Code Templates → Code → New Java Files → Edit… →

```
/*
 * (C) Copyright ${year} IUH. All rights reserved.
 *
 * @author: ${user}
 * @date: ${date}
 * @version: 1.0
 */
```

Installation

```
<dependencies>
        <!-- https://mvnrepository.com/artifact/com.google.code.gson/gson -->
        <dependency>
                <groupId>com.google.code.gson</groupId>
                <artifactId>gson</artifactId>
                <version>2.8.5</version>
        </dependency>
        <dependency>
                <groupId>org.mongodb</groupId>
                <artifactId>mongodb-driver-sync</artifactId>
                <version>4.0.5</version>
        </dependency>
        <dependency>
                <groupId>org.mongodb</groupId>
                <artifactId>mongodb-driver-legacy</artifactId>
                <version>4.0.5</version>
        </dependency>
```

```
		<!-- https://mvnrepository.com/artifact/log4j/log4j -->
		<dependency>
			<groupId>log4j</groupId>
			<artifactId>log4j</artifactId>
			<version>1.2.17</version>
		</dependency>
	</dependencies>
```

LogFactory

```
import org.apache.log4j.Logger;

public class LogFactory {
	public static Logger getLogger() {
		return Logger.getLogger(
				Thread.currentThread().getStackTrace()[2].getClassName());
	}
}
```

log4j.properties

```
# Show error log + info log + debug log
log4j.rootLogger= DEBUG, stdout, file
# Redirect log messages to console
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %p %c{1}: %C %M - %m%n

# Redirect log messages to a log file
log4j.appender.file=org.apache.log4j.RollingFileAppender
log4j.appender.file.File=logs/mylog.log
log4j.appender.file.MaxFileSize=5MB
log4j.appender.file.MaxBackupIndex=10
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n
```

Lớp đối tượng

Lophoc.java

```
import java.util.HashSet;
import java.util.Set;

public class Lophoc {
	private String malop;
	private String tenlop;
	private String ghichu;
	private Set<Sinhvien> dssv;

	/**
	 *
```

```
 */
public Lophoc() {
	this("");
}

/**
 * @param malop
 */
public Lophoc(String malop) {
	this(malop,"","");
}

/**
 * @param malop
 * @param tenlop
 * @param ghichu
 */
public Lophoc(String malop, String tenlop, String ghichu) {
	this.malop = malop;
	this.tenlop = tenlop;
	this.ghichu = ghichu;
	this.dssv = new HashSet<Sinhvien>();
}

/**
 * @return the malop
 */
public String getMalop() {
	return malop;
}
/**
 * @param malop the malop to set
 */
public void setMalop(String malop) {
	this.malop = malop;
}
/**
 * @return the tenlop
 */
public String getTenlop() {
	return tenlop;
}
/**
 * @param tenlop the tenlop to set
 */
public void setTenlop(String tenlop) {
	this.tenlop = tenlop;
}
/**
 * @return the ghichu
 */
public String getGhichu() {
```

```
		return ghichu;
	}
	/**
	 * @param ghichu the ghichu to set
	 */
	public void setGhichu(String ghichu) {
		this.ghichu = ghichu;
	}

	/**
	 * @return the dssv
	 */
	public Set<Sinhvien> getDssv() {
		return dssv;
	}

	/**
	 * @param dssv the dssv to set
	 */
	public void setDssv(Set<Sinhvien> dssv) {
		this.dssv = dssv;
	}

	@Override
	public String toString() {
		return "Lophoc [malop=" + malop + ", tenlop=" + tenlop + ", ghichu=" + ghichu + ", dssv="
+ dssv + "]";
	}

}

import java.time.LocalDate;
import java.util.List;

public class Sinhvien {
	private String mssv;
	private String hodem;
	private String ten;
	private LocalDate ngaysinh;
	private String gioitinh;
	private List<String> dsDienthoai;
	private double diemTB;

	/**
	 *
	 */
	public Sinhvien() {
	}

	/**
	 * @param mssv
```

```
 */
public Sinhvien(String mssv) {
	this.mssv = mssv;
}

/**
 * @param mssv
 * @param hodem
 * @param ten
 * @param ngaysinh
 * @param gioitinh
 * @param malop
 * @param dsDienthoai
 * @param diemTB
 */
public Sinhvien(String mssv, String hodem, String ten, LocalDate ngaysinh, String gioitinh,
		List<String> dsDienthoai, double diemTB) {
	this.mssv = mssv;
	this.hodem = hodem;
	this.ten = ten;
	this.ngaysinh = ngaysinh;
	this.gioitinh = gioitinh;
	this.dsDienthoai = dsDienthoai;
	this.diemTB = diemTB;
}
/**
 * @return the mssv
 */
public String getMssv() {
	return mssv;
}
/**
 * @param mssv the mssv to set
 */
public void setMssv(String mssv) {
	this.mssv = mssv;
}
/**
 * @return the hodem
 */
public String getHodem() {
	return hodem;
}
/**
 * @param hodem the hodem to set
 */
public void setHodem(String hodem) {
	this.hodem = hodem;
}
/**
 * @return the ten
 */
```

```
	public String getTen() {
		return ten;
	}
	/**
	 * @param ten the ten to set
	 */
	public void setTen(String ten) {
		this.ten = ten;
	}
	/**
	 * @return the ngaysinh
	 */
	public LocalDate getNgaysinh() {
		return ngaysinh;
	}
	/**
	 * @param ngaysinh the ngaysinh to set
	 */
	public void setNgaysinh(LocalDate ngaysinh) {
		this.ngaysinh = ngaysinh;
	}
	/**
	 * @return the gioitinh
	 */
	public String getGioitinh() {
		return gioitinh;
	}
	/**
	 * @param gioitinh the gioitinh to set
	 */
	public void setGioitinh(String gioitinh) {
		this.gioitinh = gioitinh;
	}
	/**
	 * @return the dsDienthoai
	 */
	public List<String> getDsDienthoai() {
		return dsDienthoai;
	}
	/**
	 * @param dsDienthoai the dsDienthoai to set
	 */
	public void setDsDienthoai(List<String> dsDienthoai) {
		this.dsDienthoai = dsDienthoai;
	}
	/**
	 * @return the diemTB
	 */
	public double getDiemTB() {
		return diemTB;
	}
	/**
```

```
	 * @param diemTB the diemTB to set
	 */
	public void setDiemTB(double diemTB) {
		this.diemTB = diemTB;
	}
	@Override
	public String toString() {
		return "Sinhvien [mssv=" + mssv + ", hodem=" + hodem + ", ten=" + ten + ", ngaysinh=" +
ngaysinh + ", gioitinh="
				+ gioitinh  + ", dsDienthoai=" + dsDienthoai + ", diemTB=" + diemTB + "]";
	}

	@Override
	public int hashCode() {
		final int prime = 31;
		int result = 1;
		result = prime * result + ((mssv == null) ? 0 : mssv.hashCode());
		return result;
	}

	@Override
	public boolean equals(Object obj) {
		if (this == obj)
			return true;
		if (obj == null)
			return false;
		if (getClass() != obj.getClass())
			return false;
		Sinhvien other = (Sinhvien) obj;
		if (mssv == null) {
			if (other.mssv != null)
				return false;
		} else if (!mssv.equalsIgnoreCase(other.mssv))
			return false;
		return true;
	}

}
```

Kết nối MongoDB

ConnectMongoDB.java

```
import com.mongodb.MongoClient;

public class ConnectMongoDB {
	private static ConnectMongoDB instance;
	private MongoClient mongoClient;
	/**
	 *
```

```
	 */
	private ConnectMongoDB() {
		mongoClient = new MongoClient();
	}

	public synchronized static ConnectMongoDB getInstance() {
		if(instance == null)
			instance = new ConnectMongoDB();
		return instance;
	}

	public MongoClient getClient() {
		return mongoClient;
	}

	public void close() {
		if(mongoClient != null)
			mongoClient.close();
	}
}
```

CRUD

CRUDLophoc.java

C: Create → insertOne, insertMany
R: Read → find, aggregate
U: Update → updateOne, updateMany, findOneAndReplace, findManyAndReplace
D: Delete → deleteOne, deleteMany, findOneAndDelete, findManyAndDelete

0/ Tạo unique index trên mã lớp, unique index trên mã số sinh viên, text index trên tên sinh viên và họ đệm
1/ Thêm lớp học và danh sách sinh viên thuộc lớp đó. Thêm thành công khi không trùng mã lớp, mã số sinh viên không trùng.
2/ Tìm lớp học khi biết mã lớp
3/ Tìm sinh viên khi biết mã số sinh viên
4/ Thêm sinh viên vào một lớp học khi biết mã lớp. Thêm thành công nếu không trùng mã số sinh viên
5/ Lấy danh sách sinh viên theo lớp học khi biết mã lớp.
6/ Tìm kiếm sinh viên theo tên hoặc họ đệm
7/ Xóa một sinh viên khi biết mã số

```
import java.util.Arrays;
import java.util.HashSet;
import java.util.Set;
```

```
import org.bson.Document;

import com.google.gson.Gson;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoCursor;
import com.mongodb.client.model.Filters;
import com.mongodb.client.model.IndexOptions;
import com.mongodb.client.model.Indexes;
import com.mongodb.client.model.Updates;
import com.mongodb.client.result.InsertOneResult;
import com.mongodb.client.result.UpdateResult;

public class CRUDLophoc {

	private static final Gson gson = new Gson();

	private MongoCollection<Document> collection;

	/**
	 * @param collection
	 */
	public CRUDLophoc(MongoCollection<Document> collection) {
		this.collection = collection;
	}

	/**
	 * Tạo unique index trên mã lớp và unique index trên mã số sinh viên
	 */
	public void createIndexes() {

		collection.createIndex(Indexes.ascending("malop"), new IndexOptions().unique(true));
		collection.createIndex(Indexes.ascending("dssv.mssv"), new IndexOptions().unique(true));
		collection.createIndex(Document.parse("{'dssv.ten':'text', 'dssv.hodem':'text'}"));

	}

	/**
	 * 1/ Thêm lớp học và danh sách sinh viên thuộc lớp đó.
	 * Thêm thành công khi không trùng mã lớp, không trùng mã số sinh viên.
	 * @param lophoc
	 * @return true, nếu thêm thành công
	 * @return false, nếu ngược lại
	 */
	public boolean themLophoc(Lophoc lophoc)  throws Exception{

		boolean result = false;

		String json = gson.toJson(lophoc);
		Document doc = Document.parse(json);
		InsertOneResult temp = collection.insertOne(doc);
		if(temp != null)
			result = true;
```

```
            return result;
        }

        /**
         * 2/ Tìm lớp học khi biết mã lớp
         * @param malop
         * @return lớp học
         */
        public Lophoc getLophoc(String malop){
            String json = "{malop:'"+malop+"'}";
            Document doc = collection.find(Document.parse(json)).first(); //Mã lớp là duy nhất
            return ( doc != null ? gson.fromJson(doc.toJson(), Lophoc.class) : null);
        }

        /**
         * 2/ Tìm lớp học khi biết mã lớp
         * @param malop
         * @return lớp học
         */
//      public Lophoc getLophoc(String malop){
//          Document doc = collection.find(Filters.eq("malop", malop)).first(); //Mã lớp là duy nhất
//          return ( doc != null ? gson.fromJson(doc.toJson(), Lophoc.class) : null);
//      }

        /**
         * 3/ Tìm sinh viên khi biết mã số sinh viên
         * db.dsLophoc.aggregate([{$project:{dssv:1, _id:0}},
         * {$unwind:'$dssv'},{$replaceWith:'$dssv'},{$match:{'mssv':'18887771'}}])
         * @param mssv
         * @return {@link Sinhvien}
         */
        public Sinhvien getSinhvien(String mssv) {

            Sinhvien sv = null;
            Document doc = collection.aggregate(Arrays.asList(
                            Document.parse("{$project:{dssv:1, _id:0}}"),
                            Document.parse("{$unwind:'$dssv'}"),
                            Document.parse("{$replaceWith:'$dssv'}"),
                            Document.parse("{$match:{'mssv':'"+mssv+"'}}"))).first();
            if(doc != null)
                sv = gson.fromJson(doc.toJson(), Sinhvien.class);

            return sv;
        }

        /**
         * 4/ Thêm sinh viên vào một lớp học khi biết mã lớp.
         * Thêm thành công nếu không trùng mã số sinh viên
         * @param malop
         * @return  <code>true</code>, nếu thêm thành công (nếu không trùng mã số sinh viên)
         * @return <code>false</code>, nếu ngược lại
```

```
     */
    public boolean themSinhvien(Sinhvien sinhvien, String malop) {

        boolean result = false;
        Document doc = collection.find(Filters.eq("dssv.mssv", sinhvien.getMssv())).first();
        if(doc == null) {
            String json = gson.toJson(sinhvien);
            Document document = Document.parse(json);
            UpdateResult temp = collection.updateOne(Filters.eq("malop", malop),
                                                     Updates.push("dssv", document));
            if(temp.getModifiedCount() > 0)
                result = true;
        }

        return result;
    }

    /**
     * 5/ Lấy danh sách sinh viên theo lớp học khi biết mã lớp.
     * db.dsLophoc.aggregate([{$match:{malop:'DHKTPM14'}},{$project:{dssv:1, _id:0}},
     * {$unwind:'$dssv'},{$replaceWith:'$dssv'}])
     * @param malop
     * @return
     */
    public Set<Sinhvien> getDSSinhvien(String malop) {
        Set<Sinhvien> list = new HashSet<Sinhvien>();
        MongoCursor<Document> it = collection.aggregate(Arrays.asList(
                Document.parse("{$match:{malop:'"+malop+"'}}"),
                Document.parse("{$project:{dssv:1, _id:0}}"),
                Document.parse("{$unwind:'$dssv'}"),
                Document.parse("{$replaceWith:'$dssv'}"))).iterator();
        Sinhvien sv = null;
        while(it.hasNext()) {
            Document doc = it.next();
            sv = gson.fromJson(doc.toJson(), Sinhvien.class);
            list.add(sv);
        }
        return list;
    }

    /**
     * 6/ Tìm kiếm sinh viên theo tên hoặc họ đệm
     * String indexName = col.createIndex(Document.parse("{'dssv.ten':'text', 'dssv.hodem':'text'}"));
     * db.dsLophoc.aggregate([{$match:{$text:{$search:'Minh'}}},{$project:{dssv:1, _id:0}},
     * {$unwind:'$dssv'},{$replaceWith:'$dssv'},{$match:{$or:[{ten:/Minh/},{hodem:/Minh/}]}}])
     * @param name
     * @return Set<Sinhvien>
     */
    public Set<Sinhvien> getDSSinhvienByName(String name) {
        Set<Sinhvien> dssv = new HashSet<Sinhvien>();
        MongoCursor<Document> it = collection.aggregate(Arrays.asList(
            Document.parse("{$match:{$text:{$search:'"+name+"'}}}"),
```

```
                Document.parse("{$project:{dssv:1, _id:0}}"),
                Document.parse("{$unwind:'$dssv'}"),
                Document.parse("{$replaceWith:'$dssv'}"),
                Document.parse("{$match:{$or:[{ten:/"+name+"/},{hodem:/"+name+"/}]}}")))
        .cursor();
        while(it.hasNext()) {
            Document doc = it.next();
            String json = doc.toJson();
            dssv.add(gson.fromJson(json, Sinhvien.class));
        }
        return dssv;
    }

    /**
     *  7/ Xóa một sinh viên khi biết mã số
     *  db.dsLophoc.updateMany({},{$pull:{'dssv':{mssv:'18893211'}}})
     * @param mssv
     * @return <code>true</code> nếu xóa thành công
     * @return <code>false</code> nếu ngược lại
     */
    public boolean xoaSinhvien(String mssv) {

//      Cách 1/
//      UpdateResult temp = collection.updateMany(new Document(),
//          new Document("$pull", new Document("dssv", new Document("mssv", mssv) )));

//      Cách 2/
//      UpdateResult temp = collection.updateMany(Document.parse("{}"),
//              Updates.pull("dssv", new Document("mssv", mssv)));

//      Cách 3/
        UpdateResult temp = collection.updateMany(Document.parse("{}"),
                Document.parse("{$pull:{'dssv':{mssv:'"+mssv+"'}}}"));

        return temp.getModifiedCount() > 0 ? true : false;
    }
}
```

Main Test

MainTest.java

```
package baiTapSyncDriver;

import java.time.LocalDate;
import java.util.Arrays;
import java.util.Calendar;
import java.util.HashSet;

import org.apache.log4j.Logger;
import org.bson.Document;
```

```
import com.mongodb.MongoClient;
import com.mongodb.client.MongoCollection;

public class MainTest {
	private static final Logger logger = LogFactory.getLogger();
	public static void main(String[] args) {
		MongoClient mongoClient = ConnectMongoDB.getInstance().getClient();
		 MongoCollection<Document> x =
mongoClient.getDatabase("qlsvdb").getCollection("dsLophoc");
		CRUDLophoc crud = new CRUDLophoc(x  );

//		0/Tao unique index trên mã lớp và unique index trên mã số sinh viên
		crud.createIndexes();

//		1/ Thêm lớp học và danh sách sinh viên thuộc lớp đó. Thêm thành công khi không trùng mã
lớp, mã số sinh viên không trùng.

		Lophoc lophoc = new Lophoc("DHKHMT16K", "Đai học khoa học máy tính 16K", "Ngành
Khoa học máy tính");
		lophoc.setDssv(new HashSet<Sinhvien>(Arrays.asList(new Sinhvien("18777771", "Nguyễn
Văn", "An", LocalDate.of(2000, Calendar.SEPTEMBER, 11), "Nam",
				Arrays.asList("0913444555"), 10))));

		boolean ketqua = false;
		try {
			ketqua = crud.themLophoc(lophoc);
		}catch (Exception e) {
			logger.info(e.getMessage());
		}
		if(ketqua)
			System.out.println("Thêm thành công!");
		else
			System.out.println("Thêm không thành công!");

//		2/ Tìm lớp học khi biết mã lớp
//		Lophoc lophoc = crud.getLophoc("DHKHMT16K");
//		if(lophoc != null)
//			System.out.println(lophoc);
//		else
//			System.out.println("Không tìm thấy!");

//		 3/ Tìm sinh viên khi biết mã số sinh viên
//		Sinhvien sv = crud.getSinhvien("18351321");
//		if(sv != null)
//			System.out.println(sv);
//		else
//			System.out.println("Không tìm thấy!");

//		4/ Thêm sinh viên vào một lớp học khi biết mã lớp. Thêm thành công nếu không trùng mã số
sinh viên
```

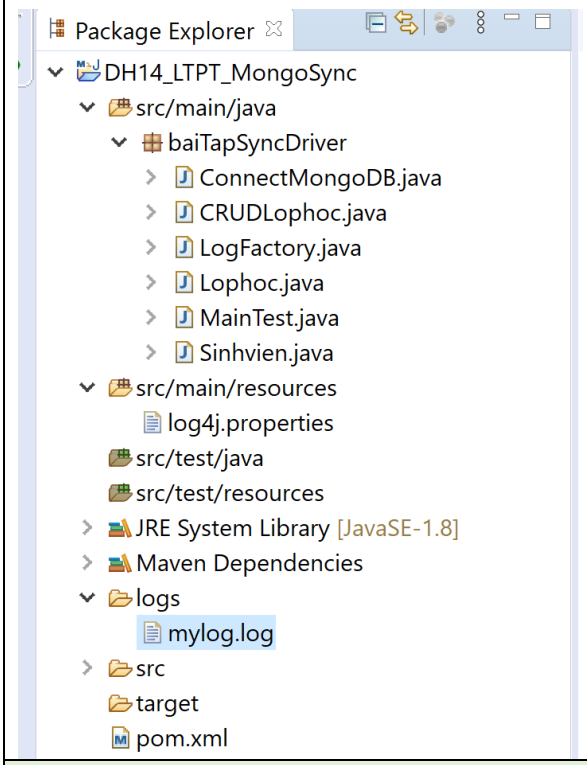
```
//		Sinhvien sv = new Sinhvien("18889991", "Nguyễnh Văn", "Anh", LocalDate.of(2000,
Calendar.SEPTEMBER, 11), "Nam", Arrays.asList("0913444555"), 10);
//		boolean kq = crud.themSinhvien(sv, "DHKHMT16K");
//		if(kq)
//			System.out.println("Thêm thành công!");
//		else
//			System.out.println("Thêm không thành công!");

//		5/ Lấy danh sách sinh viên theo lớp học khi biết mã lớp.
//		Set<Sinhvien> dssv = crud.getDSSinhvien("DHKHMT16K");
//		dssv.forEach(sv -> System.out.println(sv));

//		6/ Tìm kiếm sinh viên theo tên hoặc họ đệm
//		Set<Sinhvien> dssv = crud.getDSSinhvienByName("Đức");
//		dssv.forEach(sv -> System.out.println(sv));

//		 7/ Xóa một sinh viên khi biết mã số
//		boolean result = crud.xoaSinhvien("18351321");
//		if(result)
//			System.out.println("Đã xóa!");
//		else
//			System.out.println("Xóa không thành công!");
	}
}
```

Cấu trúc chương trình



Full Source Code

http://www.mediafire.com/file/6jg5b5wrbqgh1wg/DH14_LTPT_MongoSync.rar