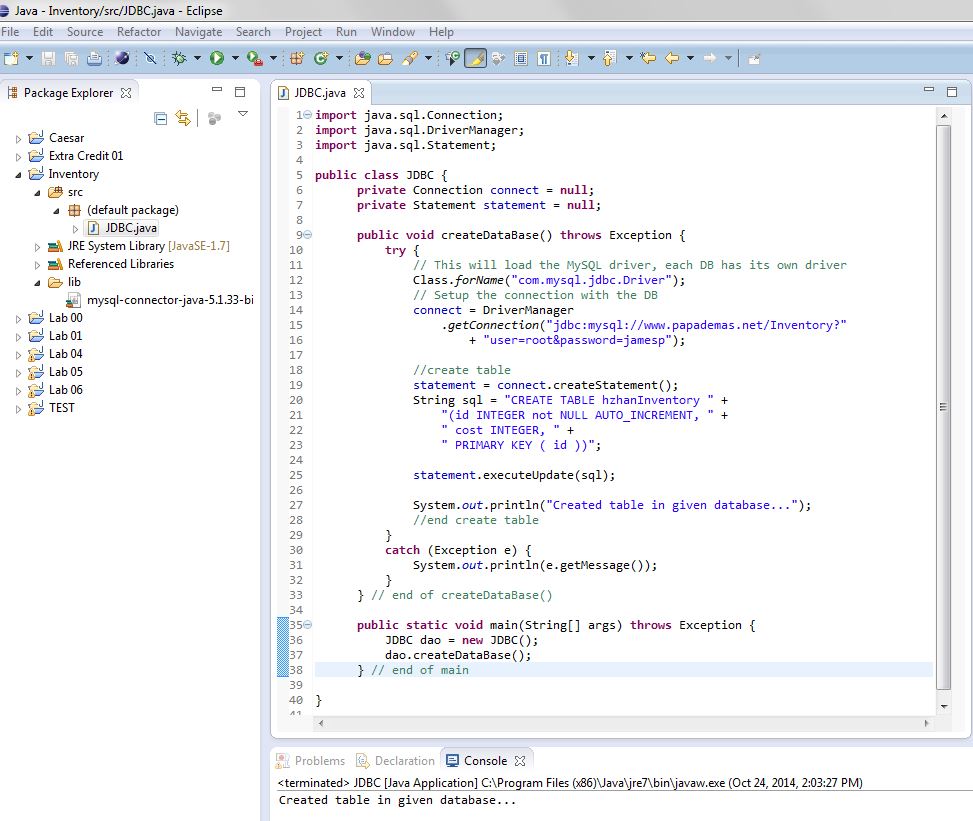
**Step 5 Create a table in database:**

**Sample Output:**



**Setp 6 Insert records into my table:**

**Source code:**

**import** java.sql.Connection;

**import** java.sql.DriverManager;

**import** java.sql.Statement;

**public** **class** JDBC {

**private** Connection connect = **null**;

**private** Statement statement = **null**;

**public** **void** createDataBase() **throws** Exception {

**try** {

// This will load the MySQL driver, each DB has its own driver

Class.*forName*("com.mysql.jdbc.Driver");

// Setup the connection with the DB

connect = DriverManager

.*getConnection*("jdbc:mysql://www.papademas.net/Inventory?"

+ "user=root&password=jamesp");

//create table

statement = connect.createStatement();

String sql = "CREATE TABLE hzhanInventory " +

"(id INTEGER not NULL AUTO\_INCREMENT, " +

" cost INTEGER, " +

" PRIMARY KEY ( id ))";

statement.executeUpdate(sql);

System.*out*.println("Created table in given database...");

//end create table

}

**catch** (Exception e) {

System.*out*.println(e.getMessage());

}

} // end of createDataBase()

**public** **void** insertIntoDataBase() **throws** Exception {

**try** {

// This will load the MySQL driver, each DB has its own driver

Class.*forName*("com.mysql.jdbc.Driver");

// Setup the connection with the DB

connect = DriverManager

.*getConnection*("jdbc:mysql://www.papademas.net/Inventory?"

+ "user=root&password=jamesp");

System.*out*.println("Inserting records into the table...");

statement = connect.createStatement();

String sql = "INSERT INTO hzhanInventory(cost) " +

"VALUES (400)";

statement.executeUpdate(sql);

sql = "INSERT INTO hzhanInventory(cost) " +

"VALUES (400)";

statement.executeUpdate(sql);

sql = "INSERT INTO hzhanInventory(cost) " +

"VALUES (500)";

statement.executeUpdate(sql);

sql = "INSERT INTO hzhanInventory(cost) " +

"VALUES (500)";

statement.executeUpdate(sql);

sql = "INSERT INTO hzhanInventory(cost) " +

"VALUES (600)";

statement.executeUpdate(sql);

System.*out*.println("Inserted records into the table...");

}

**catch** (Exception e) {

System.*out*.println(e.getMessage());

}

} // end of insertIntoDataBase()

**public** **static** **void** main(String[] args) **throws** Exception {

JDBC dao = **new** JDBC();

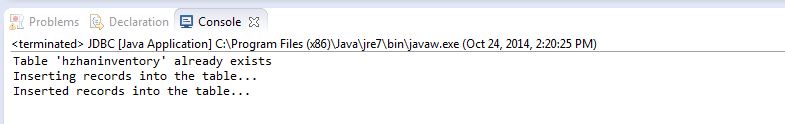
dao.createDataBase();

dao.insertIntoDataBase();

} // end of main

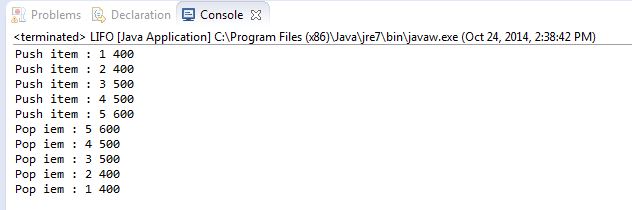
}

**Sample Output:**



**Step 7 Show the stacks’ pushed and popped values**

**Sample Output:**



**Step 8 Modify LIFO class**

**Source code:**

/\*

Program to To create a database,

insert records into it then retrieve

data from it for evaluation.

Programmer: Hong Zhang,

File Name: LIFO.java

\*/

**import** java.sql.DriverManager;

**import** java.sql.ResultSet;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.text.DecimalFormat;

**import** java.util.Scanner;

**import** com.mysql.jdbc.Connection;

**import** com.mysql.jdbc.Statement;

**public** **class** LIFO <T> {

**private** **static** Connection *connect* = **null**;

**private** **static** Statement *statement* = **null**;

**private** **static** ResultSet *resultSet* = **null**;

**private** **static** **int** *count*;

**private** T[] data;

**public** LIFO() {

data = (T[]) **new** Object[5];

*count* = 0;

} // end of public LIFO()

**void** expandCapacity() {

data = Arrays.*copyOf*(data, data.length \* 2);

} // end of expandCapacity()

**void** push(T e) {

**if** (*count* == data.length)

expandCapacity();

data[*count*++] = e;

} // end of push

T pop() **throws** Exception {

**if** (*count* <= 0) {

**throw** **new** Exception("stack empty");

}

*count*--;

**return** data[*count*];

} // end of pop()

**boolean** isEmpty() {

**return** *count* == 0;

} // end of isEmpty()

**static** **int** size() {

**return** *count*;

} // end of size()

**public** **static** **void** main(String[] args) **throws** Exception {

LIFO<Integer> s = **new** LIFO<Integer>();

**try** {

// This will load the MySQL driver, each DB has its own driver

Class.*forName*("com.mysql.jdbc.Driver");

// Setup the connection with the DB

*connect* = (Connection) DriverManager .*getConnection*("jdbc:mysql://www.papademas.net/Inventory?"

+ "user=root&password=jamesp");

// Statements allow to issue SQL queries to the database

*statement* = (Statement) *connect*.createStatement();

// Result set gets the result of the SQL query

*resultSet* = *statement*

.executeQuery("select cost from hzhanInventory");

// ResultSet is initially before the first data set

**while** (*resultSet*.next()) {

/\* column data may be retrieved via name

e.g. resultSet.getString("name");

or via the column number which starts at 1

e.g. resultSet.getString(1); \*/

**int** cost = *resultSet*.getInt(1); //retrieve cost

s.push(cost); //push cost value onto stack

}

// local variables

ArrayList<Integer> itemcost = **new** ArrayList<Integer>();

**int** number = 0;

**int** totalcost = 0;

**int** solditems = 0;

**int** soldprice = 0;

**int** soldcost = 0;

**int** soldamount = 0;

**int** profit = 0;

**int** endcost = 0;

**double** turnover = 0.00;

// declare a Scanner class object

Scanner sc = **new** Scanner(System.***in***);

// declare a DecimalFormat class object

DecimalFormat two = **new** DecimalFormat("0.00");

**while** (!s.isEmpty())//pop values from stack

{

number = s.pop();

totalcost += number;

itemcost.add(number);

}

// convert arrayList to array

**int**[] items = **new** **int**[itemcost.size()];

**for** (**int** i = 0; i < items.length; i++)

{

items[i] = itemcost.get(i).intValue();

}

// input items which were sold

System.***out***.println("There are " + items.length + " items in stock.");

System.***out***.print("please input the number of sold items: ");

solditems = sc.nextInt();

// input item price which was sold

System.***out***.print("please input the unit price of sold items: ");

soldprice = sc.nextInt();

System.***out***.println();

System.***out***.println("LIFO Method: ");

// Determine the Goods Sold (COGS) under the LIFO method

System.***out***.printf("%-30s", "Cost of Goods Sold:" );

**for** (**int** j = 0; j < (solditems - 1); j++) {

soldcost += items[j];

System.***out***.print(items[j] + " + " );

}

soldcost += items[solditems - 1];

System.***out***.println(items[solditems - 1] + " = " + soldcost);

// Determine the Profit under the LIFO method

soldamount = solditems \* soldprice;

profit = soldamount - soldcost;

System.***out***.printf("%-30s", "Profit:" );

System.***out***.println(soldamount + " - " + soldcost + " = " + profit);

// Determine the Ending Inventory under the LIFO method

System.***out***.printf("%-30s", "End Inventory:" );

**if** (solditems == items.length)

System.***out***.println(0);

**else** {

**for** (**int** l = solditems; l < (items.length - 1); l++) {

endcost += items[l];

System.***out***.print(items[l] + " + " );

}

endcost += items[items.length - 1];

System.***out***.println(items[items.length - 1] + " = " + endcost);

}

// Determine also the Inventory Turnover

**int** averagecost = (totalcost + endcost) / 2;

System.***out***.printf("%-30s", "Average Inventory:" );

System.***out***.println("(" + totalcost + " + " + endcost + ")" +

" / " + 2 + " = " + averagecost);

turnover = soldcost / (**double**)(averagecost);

System.***out***.printf("%-30s", "Inventory Turnover:" );

System.***out***.println(soldcost + " / " + averagecost + " = "

+ two.format(turnover));

// dismiss the Scanner class object

sc.close();

}

**catch** (Exception e) {

System.***out***.println(e.getMessage());

}

} // end of main

} // public class LIFO <T>

**Sample Output:**

