DUE DATE: **Mon/Tue, April 13/14 2020 during Lab time**

Introduction to Systems Programming (System I)

Lab #7

Max Points: 50

Objective: The objective of this Laboratory exercise is to:

- Gain experience with interfacing a C++ program and MySQL RDBMS
- Create a simple web-application in C++

Fill in answers to all of the questions. For some of the questions you can simply copy-paste appropriate text from the terminal/output window into this document. You may discuss the questions with your instructor.

Name: **Ben Hilger**

Part #1: Understanding with Bind variables

Estimated time: 12 minutes

Background: Developing programs that interface with a relational database often involves using user-input for querying the database – i.e., the user input needs to be supplied to SQL statements being run by the application. Bind variables provides this feature. Bind variables are placeholders in a SQL statement and are coded as `%0`, `%1`, etc. They are not actual values but serve as placeholders where the actual user-input will be substituted (*generally known as macro substitution in the CS literature*). The actual inputs are supplied when the `store` method is invoked.

Exercise:
Given the following SQL statements involving bind variables, answer the following questions. The first question has been completed for you to illustrate an example.

1. Given the following C++ and SQL statements indicate the number of bind variables and the final/actual SQL statement that is run by the database.

```
mysqlpp::Query query = myDB.query();
query << "SELECT pname, price, category, manufacturer FROM Product "
      << "WHERE price <= %0;";
query.parse();
query.store(15);
```

How many bind variables are used?	1

Show the actual SQL executed by the database:

```
SELECT pname, price, category, manufacturer FROM Product
WHERE price <= 15;
```

2. Given the following C++ and SQL statements indicate the number of bind variables and the final/actual SQL statement that is run by the database.

```
mysqlpp::Query query = myDB.query();
query << "SELECT pname, price, category, manufacturer FROM Product "
      << "WHERE price <= %1 AND price > %0;";
query.parse();
query.store(0.99, 99.99);
```

How many bind variables are used? 2

Show the actual SQL executed by the database:

```
 SELECT pname, price, category, manufacturer FROM Product
 WHERE price <= 99.99 AND price > 0.99;
```

3. Given the following C++ and SQL statements indicate the number of bind variables and the final/actual SQL statement that is run by the database.

```
mysqlpp::Query query = myDB.query();
query << "SELECT pname, price, category, manufacturer FROM Product "
      << "WHERE pname LIKE '%%%0%%' AND manufacturer = '%%%1';";
query.parse();
query.store("Gadget", "an");
```

How many bind variables are used? 2

Show the actual SQL executed by the database:

```
 SELECT pname, price, category, manufacturer FROM Product
WHERE pname LIKE '%%Gadget%%' AND manufacturer = '%%an';
```

Part #2: Setting up starter C++ code

Estimated time: 10 minutes

Objective: The objective of this part of the exercise is to setup a standard Miami University C++ project on `os1.csi.miamiOH.edu` server.

Procedure: Setup a C++ project in the following manner:

1. Using `scp`, Copy-paste the starter C++ code (`exercise11.cpp`) for this exercise appropriately into your project. Also copy across the supplied `ex11.html` (you will use it later in this exercise)

2. Study the C++ source code to ensure you understand its operations.

3. Compile and run the program. It should compile and run correctly to produce the following output:

```
MultiTouch  204   Household   Hitachi
SingleTouch 150   Photography Canon
Powergizmo  30    Gadgets     GizmoWorks
Gizmo 20    Gadgets     GizmoWorks
```

Part #3: Extend the program to work with bind variables

Estimated time: 15 minutes

Objective: The objective of this part of the exercise is to extend the starter code to:

- Accept price as 1-line of input from the user in the format `price=int`, where `int` is a number (*e.g.*, `price=10`)
- Modify the program to use a bind variable and print all products in the database whose price is less-than-or-equal to the given value.

Exercise: Complete this exercise via the following procedure

1. Modify the program to read 1-line (string) of input from the user. **No prompts needed**.
2. Extract the price value from the input string using `substr` and convert it to an integer using `std::stoi`.
3. Modify the SQL to use a bind variable.
4. Modify the program to supply value for the bind variable.
5. Test operation of the program.

Sample inputs and outputs:

User inputs are shown in bold

```
price=120
Powergizmo                  30   Gadgets     GizmoWorks
Gizmo                       20   Gadgets     GizmoWorks
```

Part #4: Extend the program to print HTML formatted output

Estimated time: 15 minutes

Background: Viewing large table outputs can be a bit cumbersome. Consequently, formatting it as an HTML table is convenient. An HTML table format consists of the following markup –

```
<table>
<tr><td>1</td><td>2</td>
<tr><td>3</td><td>4</td>
…
</table>
```

- A table begins with `<table>` and ends with `</table>`
- Each row in a table begins with `<tr>` and ends with `</tr>`
- Each column in each row begins with `<td>` and ends with `</td>`

Exercise: Extend the program from previous part to print output in HTML format as shown in the sample output while noting the following.

- **Note**: This is a trivial printing task.
- **Note**: The first Content-Type line (that indicates output is in HTML) is printed as: `std::cout << "Content-Type: text/html\r\n\r\n";`

Sample inputs and outputs:
User inputs are shown in bold. This is the same output from previous part but has HTML tags around each row and column.

```
price=120
Content-Type: text/html

<table border=1>
<tr><td>Powergizmo</td><td>30</td><td>Gadgets</td><td>GizmoWorks</td></tr>
<tr><td>Gizmo</td><td>20</td><td>Gadgets</td><td>GizmoWorks</td></tr>
</table>
```

Part #5: Using your C++ program as a web-application

Estimated time: 15 minutes

Background: A suitably designed C++ executable (not source code) can be directly used to operate as a web-application – i.e., it can accept inputs from a suitably designed HTML page and produce outputs. This operation is facilitated by web-servers, particularly the `Apache` web-server setup on `os1.csi`.

Exercise: This part of the exercise just requires you to create a directory and copy files for testing. Setup your web-application via the following procedure:

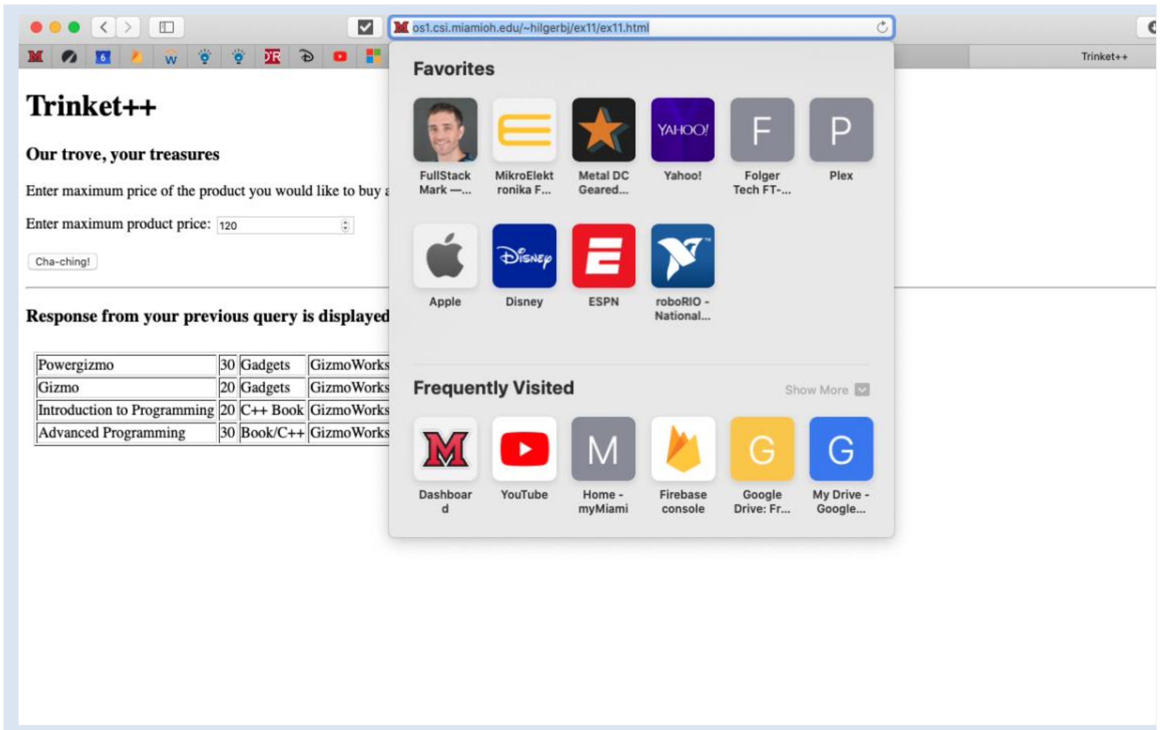
1. Open a terminal and log onto `os1.csi.miamioh.edu`.
2. Change to your `exercise11` NetBeans project directory. Run the `ls` command to ensure you are in the correct directory.
3. From your project directory run the following commands Note: These commands must be executed from within the `NetBeans` project directory. Of course you should be able to explain what the following 3 commands are doing:

```
$ mkdir ~/public_html/ex11
$ cp exercise11 ~/public_html/ex11/ex11.cgi
$ cp ex11.html ~/public_html/ex11
```

4. Now double-check your setup via the `ls` command shown below. Your setup (with 2 files) should appear exactly as shown below:

```
$ ls ~/public_html/ex11/
ex11.cgi  ex11.html
```

5. Viola! You now have a store on the web. Congratulations. Try it out via `http://os1.csi.miamioh.edu/~MUID/ex11/` where MUID is your Miami ID.
6. Place a screenshot of your website (showing your URL and a sample output) in the space below:



Trinket++

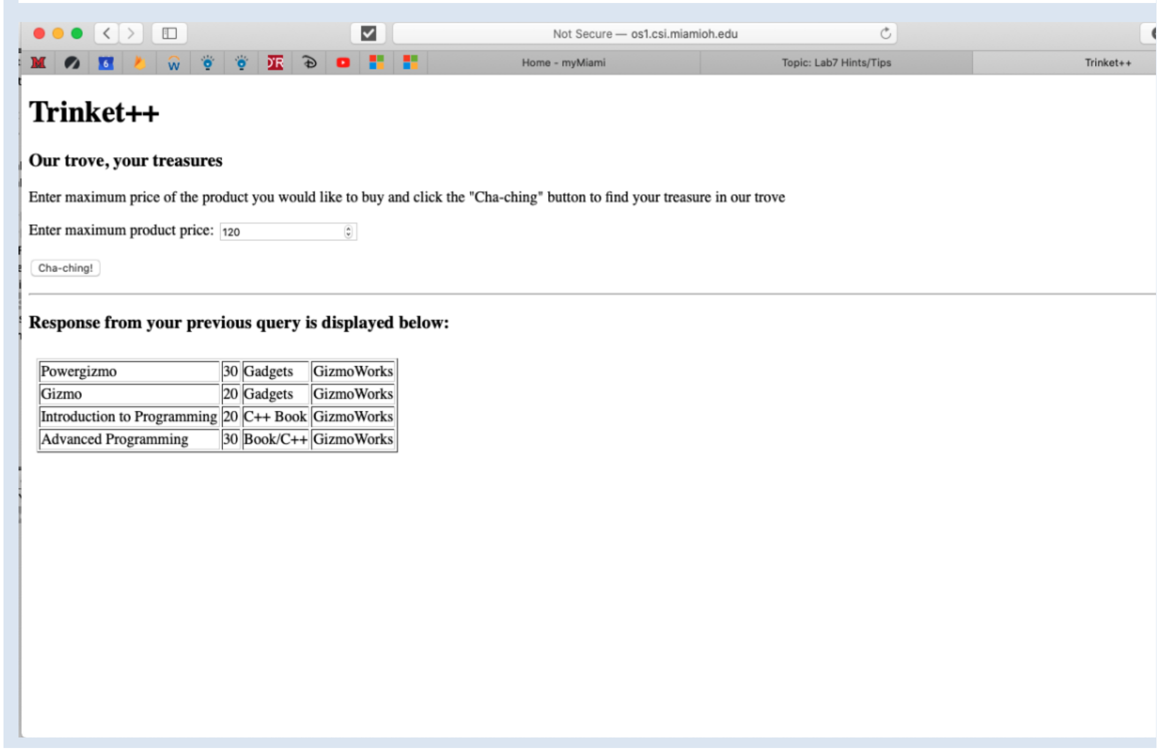
Our trove, your treasures

Enter maximum price of the product you would like to buy a

Enter maximum product price: 120

Response from your previous query is displayed

Powergizmo	30	Gadgets	GizmoWorks
Gizmo	20	Gadgets	GizmoWorks
Introduction to Programming	20	C++ Book	GizmoWorks
Advanced Programming	30	Book/C++	GizmoWorks



Trinket++

Our trove, your treasures

Enter maximum price of the product you would like to buy and click the "Cha-ching" button to find your treasure in our trove

Enter maximum product price: 120

Response from your previous query is displayed below:

Powergizmo	30	Gadgets	GizmoWorks
Gizmo	20	Gadgets	GizmoWorks
Introduction to Programming	20	C++ Book	GizmoWorks
Advanced Programming	30	Book/C++	GizmoWorks

Submission

- No late assignments will be accepted!

- This work is to be done individually
- Once you successfully completed the aforementioned exercise upload the following file(s) to Canvas. This MS-Word document saved as PDF file.
- The submission file will be saved with the name ***Lab7_yourMUID*.cpp***
- Assignment is due Mon/Tue April 13/14, 2020 during Lab time
- On or before the due time, drop the *electronic copy* of your work in the *canvas*

Don't forget to Turn in the files! Lab7_yourMUID.pdf & Lab7_yourMUID*.cpp