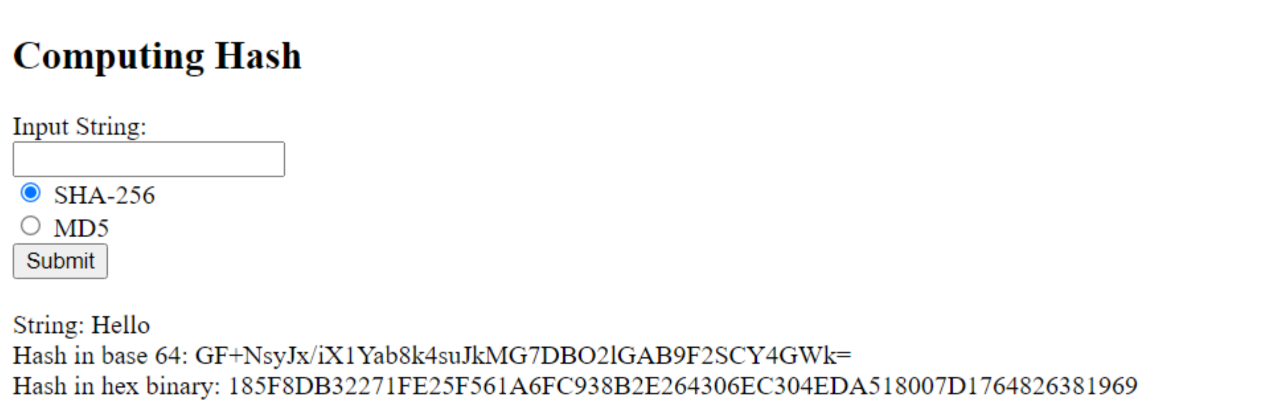
Name: Kanishka Bhambhani
Andrew Id: kbhambha

Distributed Systems: Project 1

Task 1: Computing Hash

1. Screen shots of the input and output

SHA-256



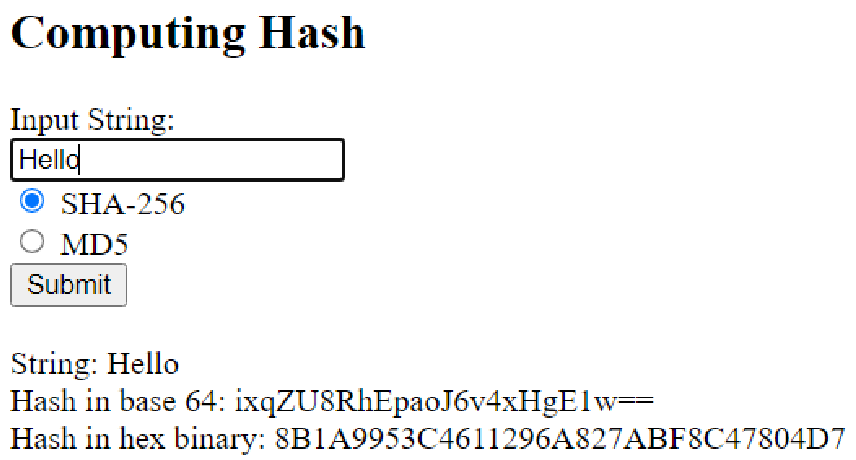
Computing Hash

Input String:

● SHA-256
○ MD5

String: Hello
Hash in base 64: GF+NsyJx/iX1Yab8k4suJkMG7DBO2lGAB9F2SCY4GWk=
Hash in hex binary: 185F8DB32271FE25F561A6FC938B2E264306EC304EDA518007D1764826381969

MD5



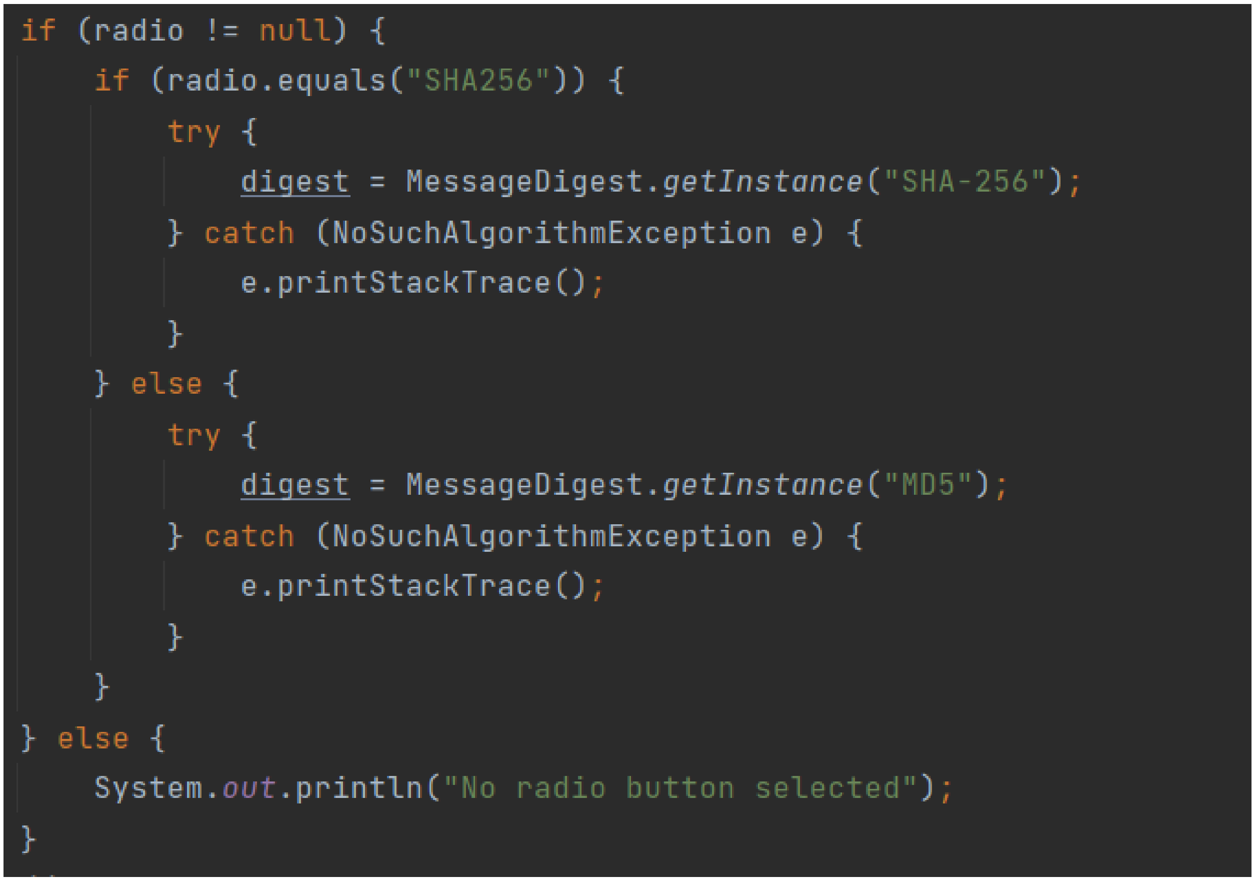
Computing Hash

Input String: Hello

● SHA-256
○ MD5

String: Hello
Hash in base 64: ixqZU8RhEpaoJ6v4xHgE1w==
Hash in hex binary: 8B1A9953C4611296A827ABF8C47804D7

2. Code Snippets:

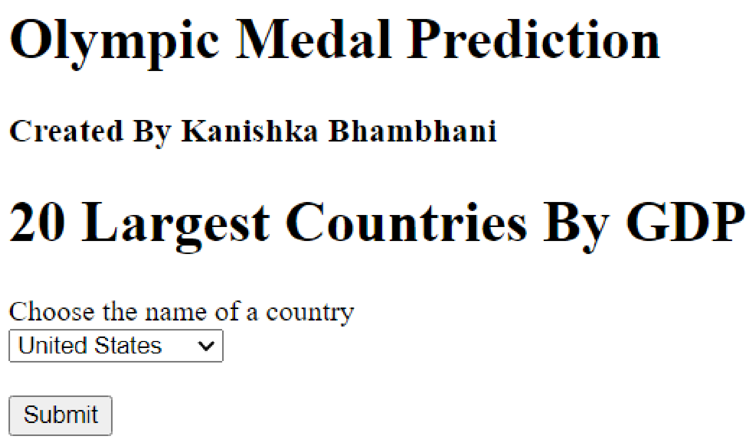


```
if (radio != null) {
    if (radio.equals("SHA256")) {
        try {
            digest = MessageDigest.getInstance("SHA-256");
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    } else {
        try {
            digest = MessageDigest.getInstance("MD5");
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    }
} else {
    System.out.println("No radio button selected");
}
```

Task 2a: Olympic Medal Prediction

1. **Screenshots:**

Input page:



Olympic Medal Prediction

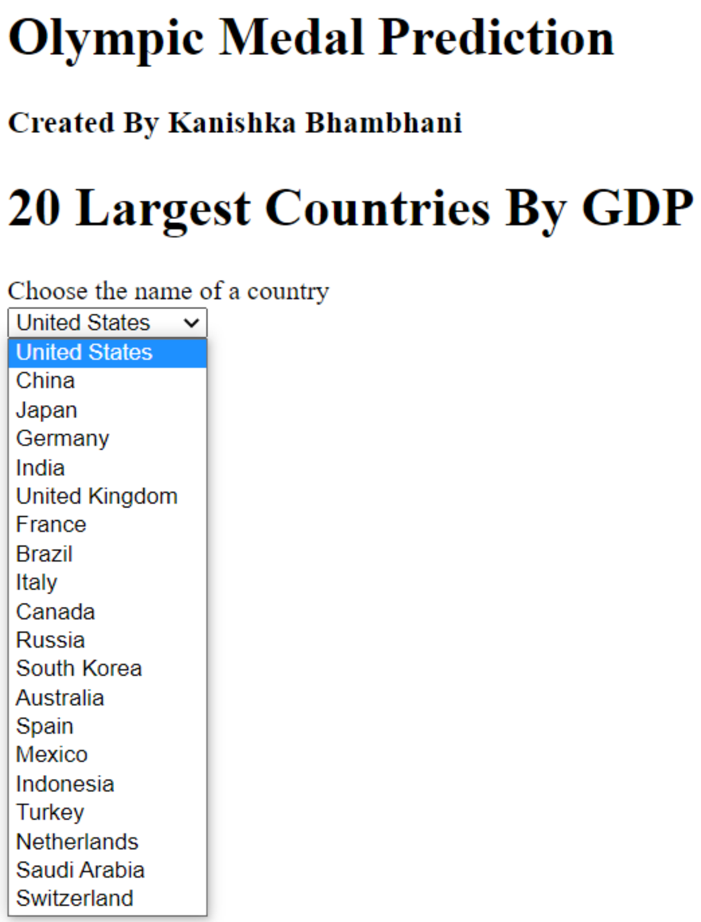
Created By Kanishka Bhambhani

20 Largest Countries By GDP

Choose the name of a country
United States

Name: Kanishka Bhambhani   Distributed Systems: Project 1
Andrew Id: kbhambha

Drop-down Menu:



Olympic Medal Prediction

Created By Kanishka Bhambhani

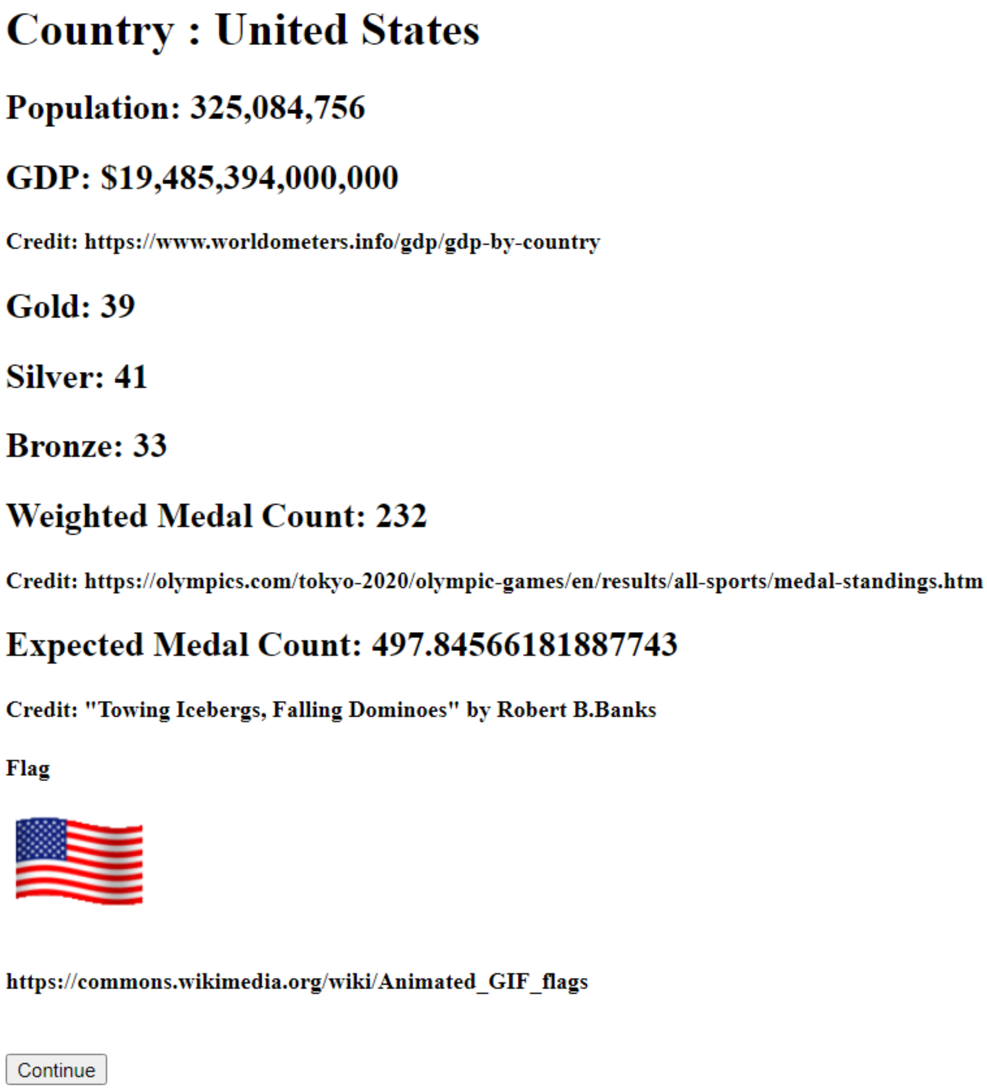
20 Largest Countries By GDP

Choose the name of a country

United States

Name: Kanishka Bhambhani Distributed Systems: Project 1
Andrew Id: kbhambha

Output for United States:



Country : United States

Population: 325,084,756

GDP: $19,485,394,000,000

Credit: https://www.worldometers.info/gdp/gdp-by-country

Gold: 39

Silver: 41

Bronze: 33

Weighted Medal Count: 232

Credit: https://olympics.com/tokyo-2020/olympic-games/en/results/all-sports/medal-standings.htm

Expected Medal Count: 497.84566181887743

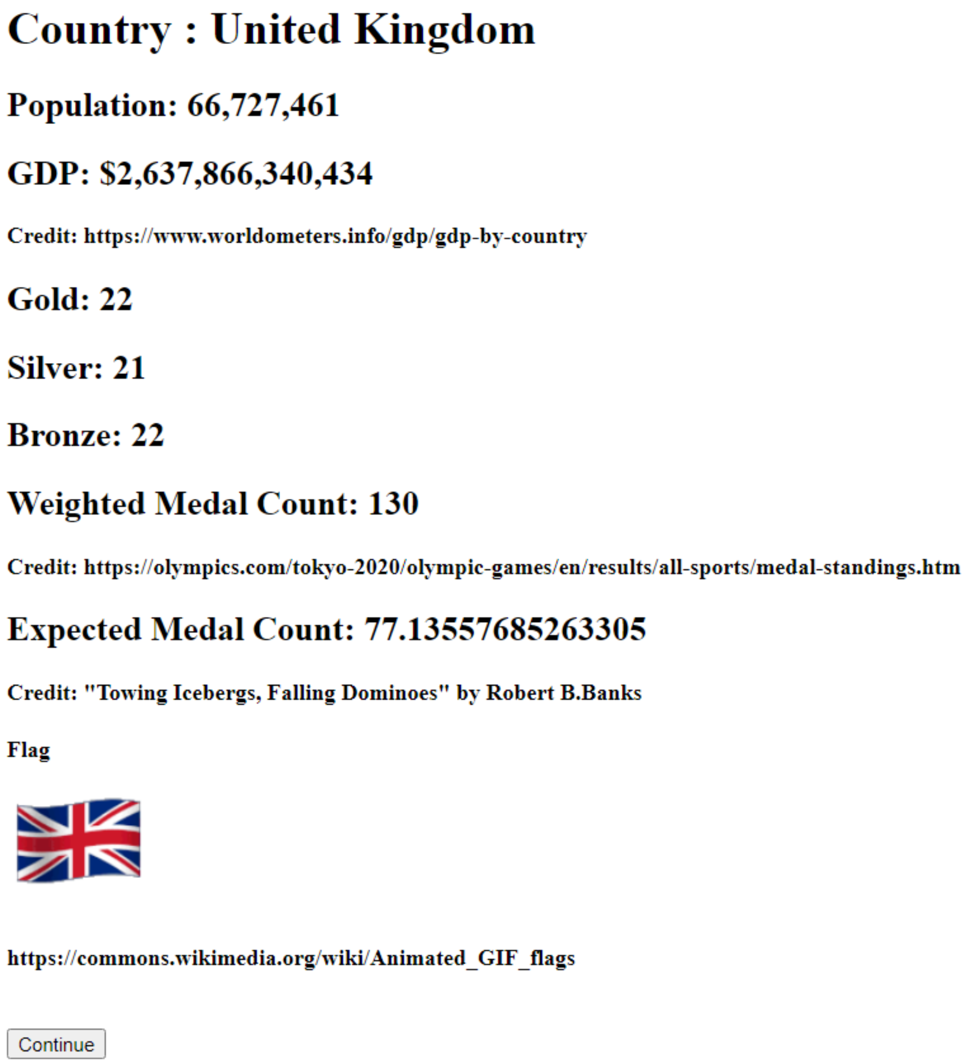
Credit: "Towing Icebergs, Falling Dominoes" by Robert B.Banks

Flag

https://commons.wikimedia.org/wiki/Animated_GIF_flags

Name: Kanishka Bhambhani Distributed Systems: Project 1
Andrew Id: kbhambha

Output for United Kingdom:



Country : United Kingdom

Population: 66,727,461

GDP: $2,637,866,340,434

Credit: https://www.worldometers.info/gdp/gdp-by-country

Gold: 22

Silver: 21

Bronze: 22

Weighted Medal Count: 130

Credit: https://olympics.com/tokyo-2020/olympic-games/en/results/all-sports/medal-standings.htm

Expected Medal Count: 77.13557685263305

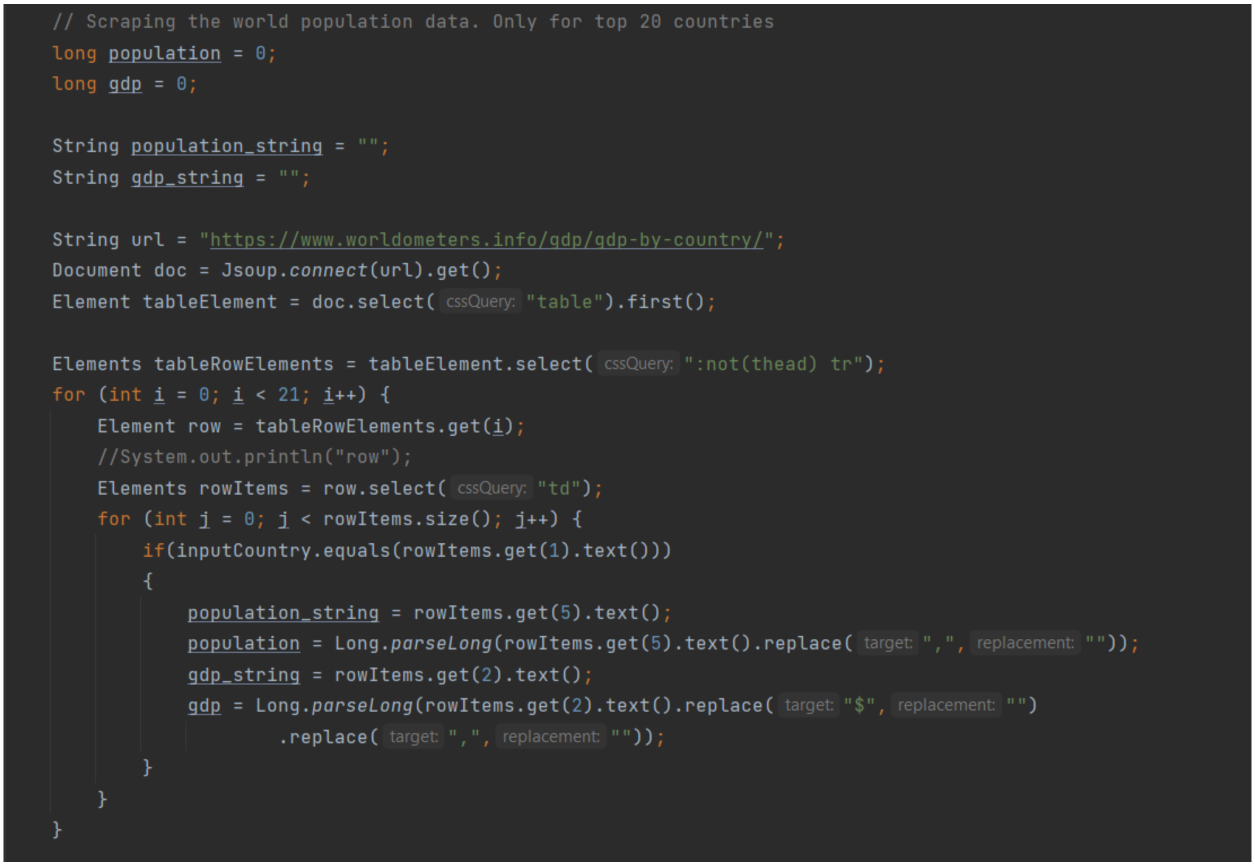
Credit: "Towing Icebergs, Falling Dominoes" by Robert B.Banks

Flag

https://commons.wikimedia.org/wiki/Animated_GIF_flags

2. Code Snippets:

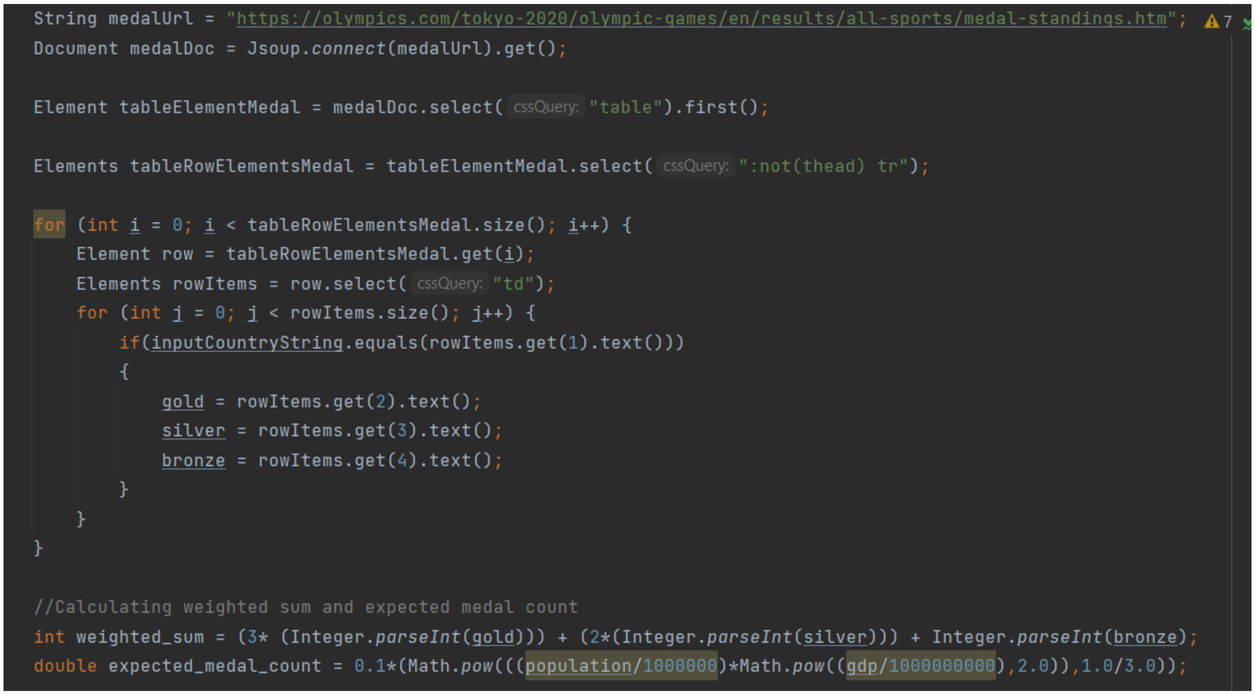
Scrapping of the population and gdp data:



```
// Scraping the world population data. Only for top 20 countries
long population = 0;
long gdp = 0;

String population_string = "";
String gdp_string = "";

String url = "https://www.worldometers.info/gdp/gdp-by-country/";
Document doc = Jsoup.connect(url).get();
Element tableElement = doc.select( cssQuery: "table").first();

Elements tableRowElements = tableElement.select( cssQuery: ":not(thead) tr");
for (int i = 0; i < 21; i++) {
    Element row = tableRowElements.get(i);
    //System.out.println("row");
    Elements rowItems = row.select( cssQuery: "td");
    for (int j = 0; j < rowItems.size(); j++) {
        if(inputCountry.equals(rowItems.get(1).text()))
        {
            population_string = rowItems.get(5).text();
            population = Long.parseLong(rowItems.get(5).text().replace( target: ",", replacement: ""));
            gdp_string = rowItems.get(2).text();
            gdp = Long.parseLong(rowItems.get(2).text().replace( target: "$", replacement: "")
                    .replace( target: ",", replacement: ""));
        }
    }
}
```

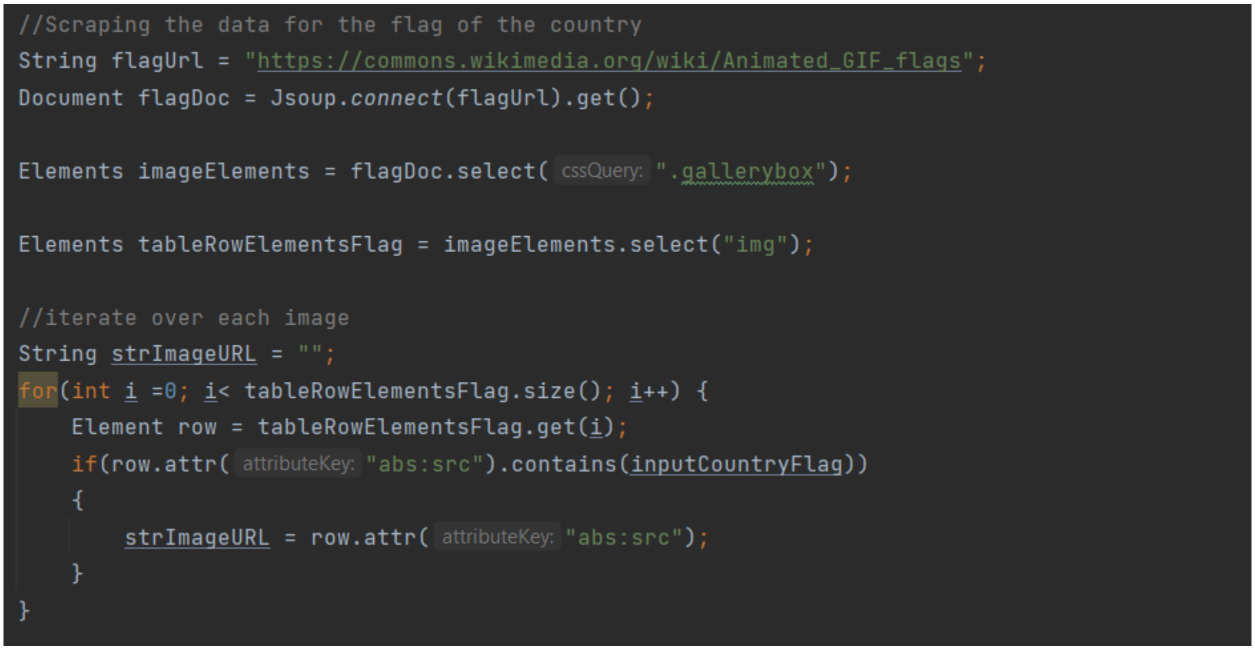
Name: Kanishka Bhambhani Distributed Systems: Project 1
Andrew Id: kbhambha

Scrapping of the medal data:



```
String medalUrl = "https://olympics.com/tokyo-2020/olympic-games/en/results/all-sports/medal-standings.htm";
Document medalDoc = Jsoup.connect(medalUrl).get();

Element tableElementMedal = medalDoc.select( cssQuery: "table").first();

Elements tableRowElementsMedal = tableElementMedal.select( cssQuery: ":not(thead) tr");

for (int i = 0; i < tableRowElementsMedal.size(); i++) {
    Element row = tableRowElementsMedal.get(i);
    Elements rowItems = row.select( cssQuery: "td");
    for (int j = 0; j < rowItems.size(); j++) {
        if(inputCountryString.equals(rowItems.get(1).text()))
        {
            gold = rowItems.get(2).text();
            silver = rowItems.get(3).text();
            bronze = rowItems.get(4).text();
        }
    }
}

//Calculating weighted sum and expected medal count
int weighted_sum = (3* (Integer.parseInt(gold))) + (2*(Integer.parseInt(silver))) + Integer.parseInt(bronze);
double expected_medal_count = 0.1*(Math.pow(((population/1000000)*Math.pow((gdp/1000000000),2.0)),1.0/3.0));
```

Scrapping of flag data:

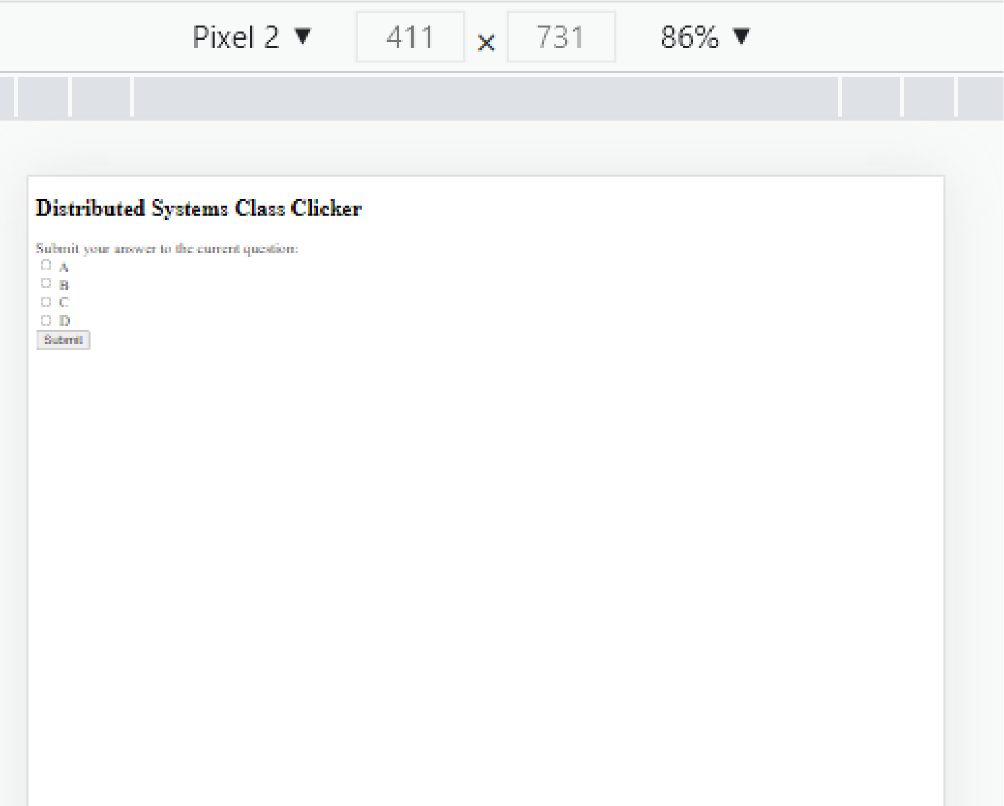


```
//Scraping the data for the flag of the country
String flagUrl = "https://commons.wikimedia.org/wiki/Animated_GIF_flags";
Document flagDoc = Jsoup.connect(flagUrl).get();

Elements imageElements = flagDoc.select( cssQuery: ".gallerybox");

Elements tableRowElementsFlag = imageElements.select("img");

//iterate over each image
String strImageURL = "";
for(int i =0; i< tableRowElementsFlag.size(); i++) {
    Element row = tableRowElementsFlag.get(i);
    if(row.attr( attributeKey: "abs:src").contains(inputCountryFlag))
    {
        strImageURL = row.attr( attributeKey: "abs:src");
    }
}
```

Task 3: Distributed Systems Class Clicker

1. Screenshots of the UI:

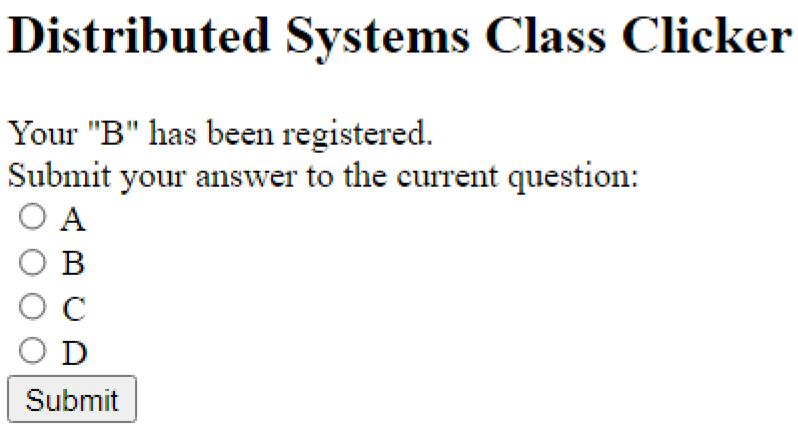
Input page in UI – on different screen size (mobile)



Distributed Systems Class Clicker

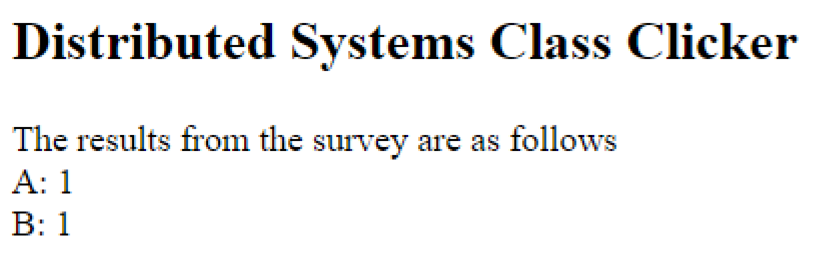
Submit your answer to the current question:
○ A
○ B
○ C
○ D
Submit

Output Page with the 1 vote


Distributed Systems Class Clicker

Your "B" has been registered.
Submit your answer to the current question:
○ A
○ B
○ C
○ D
Submit

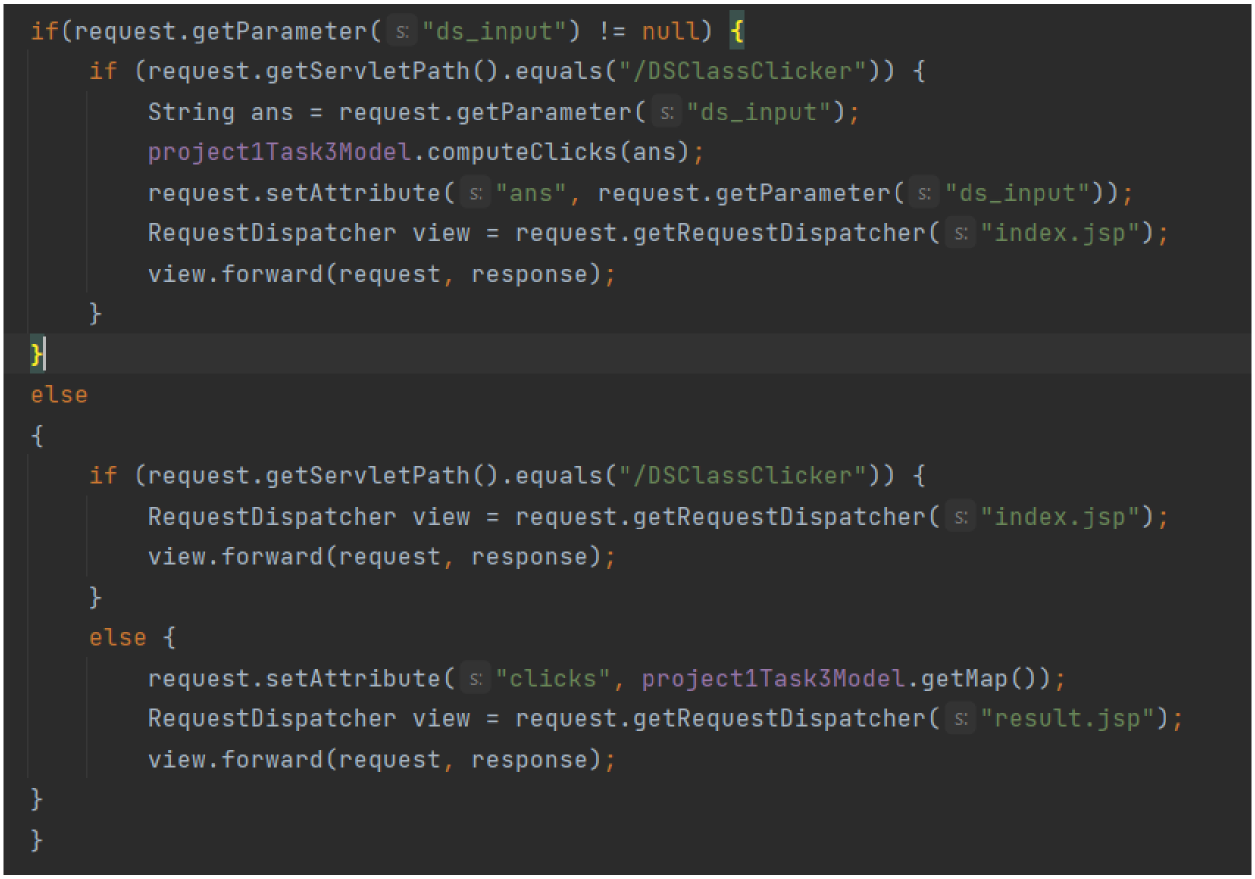
Results page



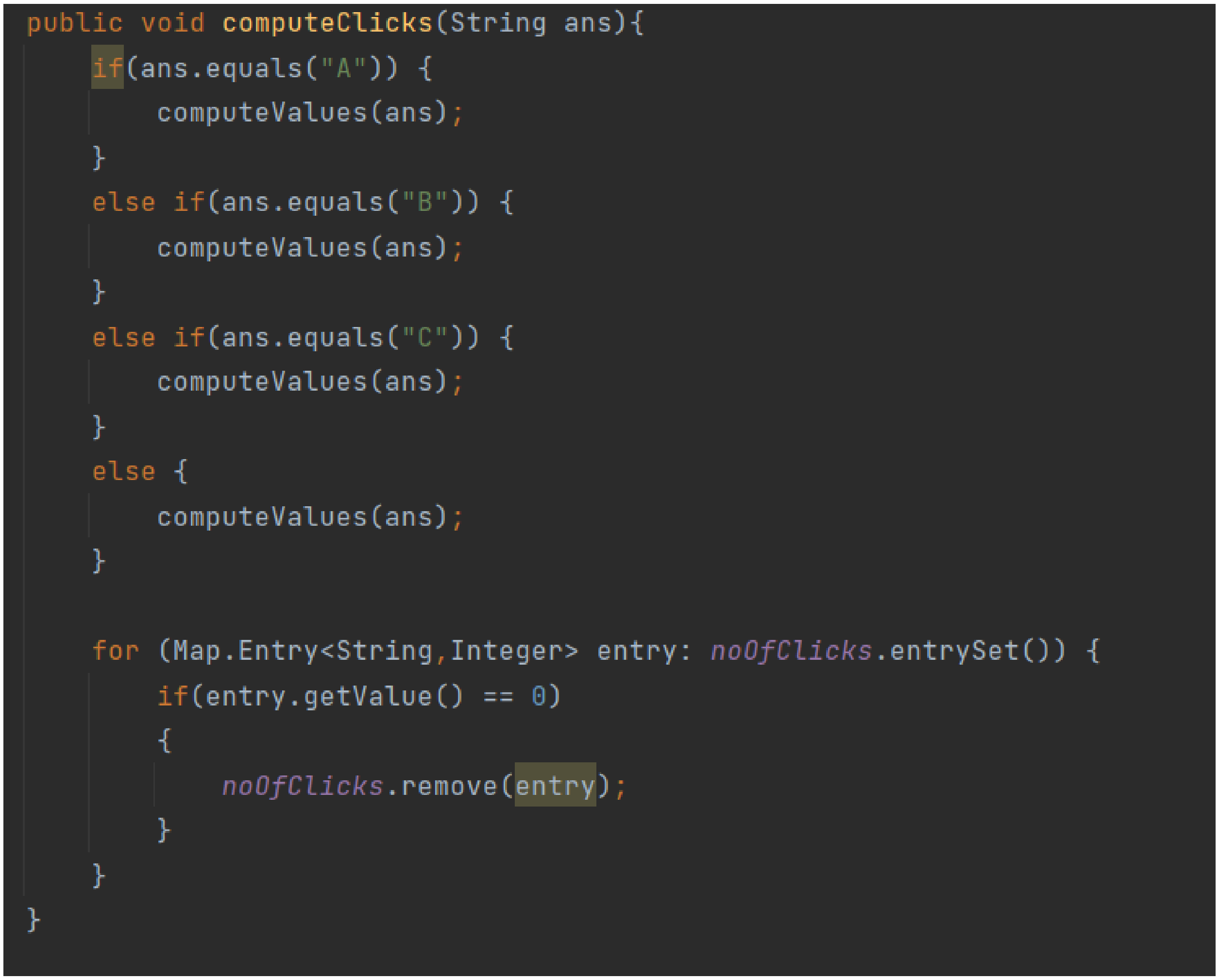
Distributed Systems Class Clicker

The results from the survey are as follows
A: 1
B: 1

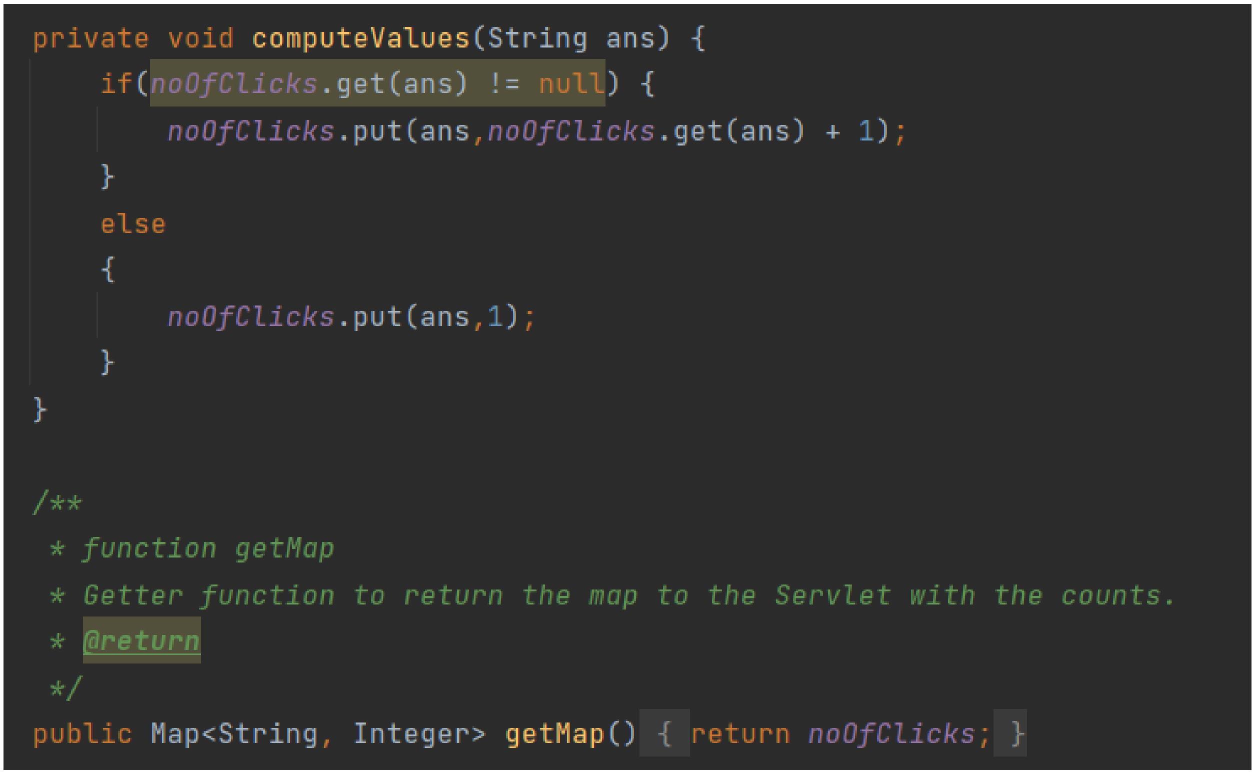
2. Code Snippet



```
if(request.getParameter( s: "ds_input") != null) {
    if (request.getServletPath().equals("/DSClassClicker")) {
        String ans = request.getParameter( s: "ds_input");
        project1Task3Model.computeClicks(ans);
        request.setAttribute( s: "ans", request.getParameter( s: "ds_input"));
        RequestDispatcher view = request.getRequestDispatcher( s: "index.jsp");
        view.forward(request, response);
    }
}
else
{
    if (request.getServletPath().equals("/DSClassClicker")) {
        RequestDispatcher view = request.getRequestDispatcher( s: "index.jsp");
        view.forward(request, response);
    }
    else {
        request.setAttribute( s: "clicks", project1Task3Model.getMap());
        RequestDispatcher view = request.getRequestDispatcher( s: "result.jsp");
        view.forward(request, response);
}
}
```



```
public void computeClicks(String ans){
    if(ans.equals("A")) {
        computeValues(ans);
    }
    else if(ans.equals("B")) {
        computeValues(ans);
    }
    else if(ans.equals("C")) {
        computeValues(ans);
    }
    else {
        computeValues(ans);
    }

    for (Map.Entry<String,Integer> entry: noOfClicks.entrySet()) {
        if(entry.getValue() == 0)
        {
            noOfClicks.remove(entry);
        }
    }
}
```



```
private void computeValues(String ans) {
    if(noOfClicks.get(ans) != null) {
        noOfClicks.put(ans,noOfClicks.get(ans) + 1);
    }
    else
    {
        noOfClicks.put(ans,1);
    }
}

/**
 * function getMap
 * Getter function to return the map to the Servlet with the counts.
 * @return
 */
public Map<String, Integer> getMap() { return noOfClicks; }
```