

# **PPL-ASSIGNMENT-BIM2015004-Q1**

AUTHOR  
Mritunjay Chaudhary



# Table of Contents

Table of contents



# Class Index

## Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<b>Boys (Class Boys )</b> .....
<b>csv_gen (Csv_gen class )</b> .....
<b>Girls (Girls class )</b> .....
<b>MyLogger (MyLogger class )</b> .....
<b>q1 (Class q1 )</b> .....

# File Index

## File List

Here is a list of all files with brief descriptions:

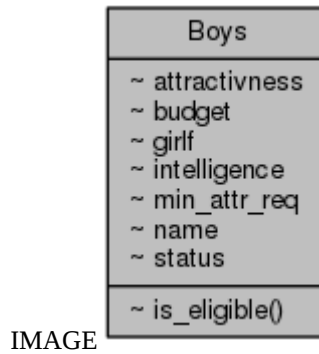
<b>Boys.java</b> .....	12
<b>csv_gen.java</b> .....	13
<b>Girls.java</b> .....	14
<b>MyLogger.java</b> .....	15
<b>q1.java</b> .....	16

# Class Documentation

## Boys

class **Boys**

Collaboration diagram for Boys:



### Package Functions

boolean **is\_eligible** (int expense, int attar)

*boolean type is\_eligible function to check conditions*

### Package Attributes

int **attractivness**

int **budget**

String **girlf**

int **intelligence**

int **min\_attr\_req**

String **name**

String **status**

---

### Detailed Description

class **Boys**

---

### Member Function Documentation

boolean **Boys.is\_eligible** (int *expense*, int *attar*)[inline], [package]

boolean type is\_eligible function to check conditions

```
13
14         {
15         if (this.budget >= expense && this.min_attr_req <= attar){
16             return true;
17         }else{
18             return false;
19         }
20     }
```

#### **Member Data Documentation**

**int Boys.attractivness[package]**

**int Boys.budget[package]**

**String Boys.girlf[package]**

**int Boys.intelligence[package]**

**int Boys.min\_attr\_req[package]**

**String Boys.name[package]**

**String Boys.status[package]**

---

**The documentation for this class was generated from the following file:**

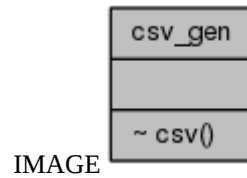
0 Boys.java



## csv\_gen

csv\_gen class

Collaboration diagram for csv\_gen:



### Package Functions

void csv ()

*csv function to generate csv files this function creates csv files and writes into it*

---

### Detailed Description

csv\_gen class

---

### Member Function Documentation

void csv\_gen.csv ()[inline], [package]

csv function to generate csv files this function creates csv files and writes into it

```
6         {
7         try{
8             FileWriter boy_file = new FileWriter("boy.csv");
9             FileWriter girl_file = new FileWriter("girl.csv");
10            int j;
11            Random boy_rand = new Random();
12            Random girl_rand = new Random();
13            for(j=0;j<=20;j++){
14
15boy_file.write("B"+j+", "+boy_rand.nextInt(10)+", "+boy_rand.nextInt(10)+", "+boy_rand.nextInt(10)+", "+boy_rand.nextInt(1000)+", "+"Single"+"\\n");
16            }
17            for(j=0;j<=7;j++){
18girl_file.write("G"+j+", "+girl_rand.nextInt(10)+", "+girl_rand.nextInt(1000)+", "+girl_rand.nextInt(10)+", "+"Single"+"\\n");
19            }
20            boy_file.close();
21            girl_file.close();
22            }catch(IOException e){
23            }
24        }
25    }
```

---

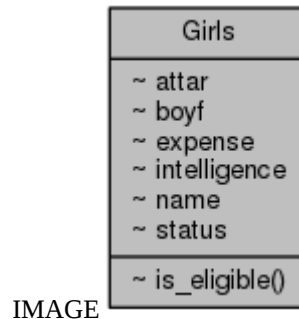
The documentation for this class was generated from the following file:

1 csv\_gen.java

## Girls

**Girls** class.

Collaboration diagram for Girls:



### Package Functions

boolean **is\_eligible** (int budget)

*boolean type is\_eligible function to check conditions*

### Package Attributes

int **attar**

String **boyf**

int **expense**

int **intelligence**

String **name**

String **status**

---

### Detailed Description

**Girls** class.

---

### Member Function Documentation

boolean **Girls.is\_eligible** (int *budget*)[inline], [package]

boolean type is\_eligible function to check conditions

```
11                                     {
12     if (this.expense <= budget) {
13         return true;
14     }
15     else {
16         return false;
17     }
18 }
```

#### **Member Data Documentation**

**int Girls.attar[package]**

**String Girls.boyf[package]**

**int Girls.expense[package]**

**int Girls.intelligence[package]**

**String Girls.name[package]**

**String Girls.status[package]**

---

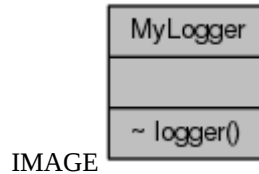
**The documentation for this class was generated from the following file:**

2   **Girls.java**

# MyLogger

**MyLogger** class.

Collaboration diagram for MyLogger:



## Static Package Functions

static void **logger** (String[] s, int cnt)

*logger function to log in to text file*

---

## Detailed Description

**MyLogger** class.

---

## Member Function Documentation

static void **MyLogger.logger** (String [] s, int cnt)[inline], [static], [package]

logger function to log in to text file

```
8                                     {
9         try {
10            Logger logger = Logger.getLogger("My log");
11            FileHandler fh = new FileHandler("log.txt", true);
12            logger.addHandler(fh);
13            SimpleFormatter formatter = new SimpleFormatter();
14            fh.setFormatter(formatter);
15            logger.setUseParentHandlers(true);
16            int k=0;
17            for(k=0;k<cnt;k++){
18                logger.info(s[k]);
19            }
20        }
21        catch (SecurityException e) {
22            e.printStackTrace();
23        }catch (IOException e) {
24            e.printStackTrace();
25        }
26    }
```

---

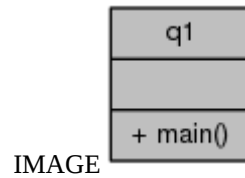
The documentation for this class was generated from the following file:

3 **MyLogger.java**

## q1

class **q1**

Collaboration diagram for q1:



### Static Public Member Functions

static void **main** (String args[])

*main class*

### Detailed Description

class **q1**

### Member Function Documentation

static void **q1.main** (String *args*[][inline], [static]

*main class*

```
5                                     {
6     int i,j;
7     i=0;
8     j=0;
9     Boys b[] = new Boys[1000];
10    csv_gen csv_files = new csv_gen();
11    csv_files.csv();
12    String csvFile = "boy.csv";
13    String line = "";
14    String csvSplit = ",";
15    BufferedReader buff = null;
16    try{
17        buff = new BufferedReader(new FileReader(csvFile));
18        while((line = buff.readLine() )!= null){
19            String[] boys_table = line.split(csvSplit);
20            // System.out.println(boy[0] + boy[1] + boy[2] + boy [3]);
21            b[i] = new Boys();
22            b[i].name = boys_table[0];
23            b[i].intelligence = Integer.parseInt(boys_table[1]);
24            b[i].attractivness = Integer.parseInt(boys_table[2]);
25            b[i].min_attr_req = Integer.parseInt(boys_table[3]);
26            b[i].budget = Integer.parseInt(boys_table[4]);
27            b[i].girlf="";
28            b[i].status = boys_table[5];
29            i++;
30        }
31    }
32
33    catch(FileNotFoundException e){
34        e.printStackTrace();
35    }catch(IOException e){
36        e.printStackTrace();
37    }
38
39    Girls g[] = new Girls[1000];
40    csvFile = "girl.csv";
```

```

41         line = "";
42         csvSplit = ",";
43         buff = null;
44
45
46         try{
47             buff = new BufferedReader(new FileReader(csvFile));
48             while((line = buff.readLine() )!= null){
49                 String[] girls_table = line.split(csvSplit);
50                 // System.out.println(boy[0] + boy[1] + boy[2] + boy [3]);
51                 g[j] = new Girls();
52                 g[j].name = girls_table[0];
53                 g[j].attar = Integer.parseInt(girls_table[1]);
54                 g[j].expense = Integer.parseInt(girls_table[2]);
55                 g[j].intelligence = Integer.parseInt(girls_table[3]);
56                 g[j].status = girls_table[4];
57                 g[j].boyf="";
58                 j++;
59             }
60         }
61         catch(FileNotFoundException e){
62             e.printStackTrace();
63         }catch(IOException e){
64             e.printStackTrace();
65         }finally{
66             if(buff != null){
67                 try{
68                     buff.close();
69                 }catch(IOException e){
70                     e.printStackTrace();
71                 }
72             }
73         }
74
75         int k,l;
76         String s[] = new String[100];
77         int cnt = 0;
78         for(k=0;k<i;k++){
79             for(l=0;l<j;l++){
80                 if(b[k].is_eligible(g[l].expense,g[l].attar) &&
g[l].is_eligible(b[k].budget) && g[l].status.equals("Single") &&
b[k].status.equals("Single") ){
81                     b[k].girlf = g[l].name;
82                     g[l].boyf = b[k].name;
83                     b[k].status = "Is_Committed";
84                     g[l].status = "Is_Committed";
85                     if(b[k].status.equals("Is_Committed")){
86                         s[cnt] = "Boy: "+ b[k].name +" is committed with
"+"Girl: "+ b[k].girlf  ;
87                     }
88
89                     cnt++;
90                 }
91             }
92         }
93     }
94 }
95
96     MyLogger.logger(s, cnt);
97 }

```

---

The documentation for this class was generated from the following file:

4   q1.java

# File Documentation

## Boys.java File Reference

### Classes

class **Boys**

*class **Boys***

## csv\_gen.java File Reference

### Classes

class **csv\_gen**  
    *csv\_gen* class



## Girls.java File Reference

### Classes

class **Girls**

*Girls* class.

## MyLogger.java File Reference

### Classes

class **MyLogger**

*MyLogger* class.

## **q1.java File Reference**

### **Classes**

class **q1**

## ***class q1* Index**

INDE