# PPL-Assignment-BIM2015004-Q2

AUTHOR
Mritunjay Chaudhary

# Class Index

## Class List

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

# File Index

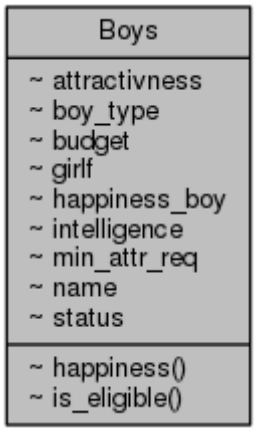
## File List

Here is a list of all files with brief descriptions:

# Class Boys

## Boys

Collaboration diagram for Boys:

IMAGE 


### Package Functions

void **happiness** (int value)
boolean **is_eligible** (int expense, int attar)

### Package Attributes

int **attractivness**
String **boy_type**
int **budget**
String **girlf**
double **happiness_boy**
int **intelligence**
int **min_attr_req**
String **name**
String **status**

---

### Detailed Description

---

### Member Function Documentation

#### void Boys.happiness (int *value*)[inline], [package]

```
21                              {
22
23          this.happiness_boy = value;
24  }
```

#### boolean Boys.is_eligible (int *expense*, int *attar*)[inline], [package]

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

```
20          }
```

---

**Member Data Documentation**

**int Boys.attractivness[package]**

**String Boys.boy_type[package]**

**int Boys.budget[package]**

**String Boys.girlf[package]**

**double Boys.happiness_boy[package]**

**int Boys.intelligence[package]**

**int Boys.min_attr_req[package]**

**String Boys.name[package]**

**String Boys.status[package]**

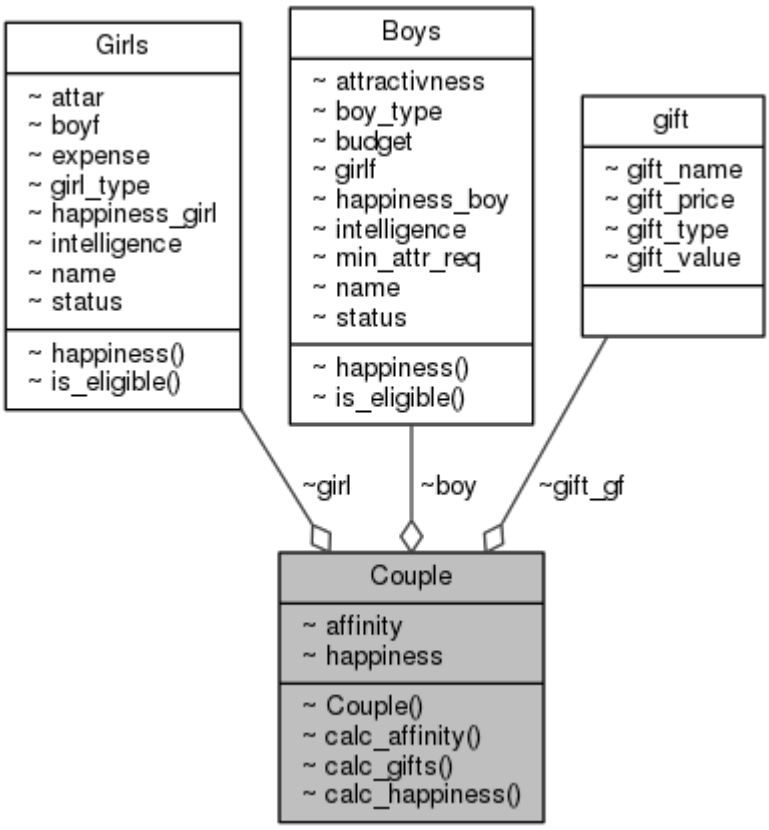
---

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

0 **Boys.java**

# Couple

Collaboration diagram for Couple:




IMAGE

### Package Functions

**Couple** (**Boys boy**, **Girls girl**)
void **calc_affinity** ()
void **calc_gifts** (**gift** gf[], int n) throws IOException
void **calc_happiness** ()

### Package Attributes

int **affinity**
**Boys boy**
**gift gift_gf** []
**Girls girl**
double **happiness** =0

---

### Detailed Description

---

### Constructor & Destructor Documentation

### Couple.Couple (Boys *boy*, Girls *girl*)[inline], [package]

```
17                                {
18          this.boy = boy;
19          this.girl = girl;
20          this.affinity = 0;
21          this.happiness = 0;
22          }
```

## Member Function Documentation

### void Couple.calc_affinity ()[inline], [package]

```
34                          {
35          int x = Math.abs(boy.budget - girl.expense);
36          int y = Math.abs(boy.attractivness - girl.attar);
37          int z = Math.abs(boy.intelligence - girl.intelligence);
38
39          this.affinity = x+y+z;
40      }
```

### void Couple.calc_gifts (gift *gf*[], int *n*) throws IOException[inline], [package]

```
41                                                          {
42
43
44
45              String df= new SimpleDateFormat("dd/MM/yy HH:mm:ss").format(new
Time(System.currentTimeMillis()));
46              String output = "";
47              this.gift_gf = new gift[50];
48              File out=new File("log.txt");
49              String type = this.boy.boy_type;
50              int gift_count = 0;
51              int i;
52              if(type.equals("Miser")){
53
54                      int sum = 0;
55
56                      for(i=0;i<n;i++){
57                          if(sum <this.girl.expense ){
58                              output = df + "-" + this.boy.name + "
--------------> "+ this.girl.name +" "+gf[i].gift_name +"\n" ;
59                              System.out.println(output);
60                              try(BufferedWriter buffer=new BufferedWriter(new
FileWriter(out,true))){
61                                  buffer.write(output);
62
63                              }
64
65
66                              sum += gf[i].gift_price;
67                              this.gift_gf[gift_count++] =gf[i];
68                          }
69                          else
70                              break;
71                  }
72                      this.boy.budget = Math.max(this.boy.budget,sum);
73                      this.boy.happiness_boy = this.boy.budget-sum;
74
75                      this.girl.happiness(gift_gf,gift_count);
76
77
78
79                      }
80
81              else if(type.equals("Generous")){
82
83                          int sum =0;
84                          int l;
85
86                          for(l=n-1;l>=0;l--){
87                              if(sum<this.boy.budget){
88
89                                  output = df + "-" + this.boy.name + "
--------------> "+ this.girl.name+" "+gf[l].gift_name  +"\n" ;
90                                  System.out.println(output);
91                                  try(BufferedWriter buffer=new
BufferedWriter(new FileWriter(out,true))){
92                                      buffer.write(output);
```

```
                                    }
                                    sum += gf[l].gift_price;
                                }
                                else
                                    break;
                            }
                            this.boy.budget = Math.max(this.boy.budget,sum);
                            this.girl.happiness(gift_gf,gift_count);
                            this.boy.happiness_boy =
(int)this.girl.happiness_girl;




            }else if(type.equals("Geeky")){

                            int sum=0;
                        //  System.out.println("hi");
                            for(i=0;i<n;i++){
                                if(sum <this.girl.expense ){

                                    output =df + "-" + this.boy.name + "
--------------> "+ this.girl.name+" "+gf[i].gift_name       +"\n" ;
                                    System.out.println(output);
                                    try(BufferedWriter buffer=new
BufferedWriter(new FileWriter(out,true))){
                                        buffer.write(output);

                                    }
                                    sum += gf[i].gift_price;
                                    this.gift_gf[gift_count++] = gf[i];
                                }
                                else
                                    break;
                        }
                            boolean flag = false;
                            this.boy.budget = Math.max(this.boy.budget,sum);
                            for(i=0;i<n;i++){
                                if ((gf[i].gift_type.equals("Luxury")) &&
(this.boy.budget- sum ) > gf[i].gift_price){
                                    for(gift p : gift_gf){
                                        if(gf[i] == p){
                                            flag = true;
                                        }
                                    }
                                    if(flag == true){
                                        continue;
                                    }
                                    else{
                                        sum += gf[i].gift_price;
                                        gift_gf[gift_count++] =gf[i];
                                        break;
                                    }
                                }
                            }

                            this.boy.happiness_boy = this.girl.intelligence;

                            this.girl.happiness(gift_gf,gift_count);



                        }


        }
```

### void Couple.calc_happiness ()[inline], [package]

```
                         {
        float mix = 1;
```

```
26          float max = 100;
27
28          Random happi = new Random();
29          float  h = happi.nextFloat()*(max-mix)+max;
30          this.happiness = h;
31
32      }
```

---

**Member Data Documentation**

**int Couple.affinity[package]**

**Boys Couple.boy[package]**

**gift Couple.gift_gf[][package]**

**Girls Couple.girl[package]**

**double Couple.happiness =0[package]**

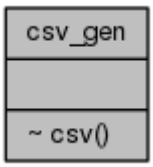
---

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

1 **Couple.java**

# csv_gen

Collaboration diagram for csv_gen:

IMAGE 


### Package Functions

void **csv** ()

---

### Detailed Description

---

### Member Function Documentation

### void csv_gen.csv ()[inline], [package]

```
 4                   {
 5         try{
 6             FileWriter boy_file = new FileWriter("boy.csv");
 7             FileWriter girl_file = new FileWriter("girl.csv");
 8             FileWriter gift_file = new FileWriter("gift.csv");
 9             int j;
10             Random boy_rand = new Random();
11             Random girl_rand = new Random();
12             Random gift_rand = new Random();
13             String boy_type[]= {"Miser","Generous","Geeky"};
14             String girl_type[]= {"Choosey","Normal","Desprate"};
15             String gift_type[] = {"Essential","Luxury","Utility"};
16 
17             for(j=0;j<=20;j++){
18 
boy_file.write("B"+j+","+boy_rand.nextInt(10)+","+boy_rand.nextInt(10)+","+boy_ran
d.nextInt(10)+","+boy_rand.nextInt(1000)+","+"Single"+","+boy_type[boy_rand.nextIn
t(3)]+"\n");
19             }
20             for(j=0;j<=8;j++){
21 
girl_file.write("G"+j+","+girl_rand.nextInt(10)+","+girl_rand.nextInt(1000)+","+gi
rl_rand.nextInt(10)+","+"Single"+","+girl_type[girl_rand.nextInt(3)]+"\n");
22             }
23             for(j=0;j<=50;j++){
24                 gift_file.write("Gift"+j +","+
gift_rand.nextInt(1000)+","+gift_rand.nextInt(10)+","+gift_type[gift_rand.nextInt(
3)]+ "\n");
25             }
26 
27             boy_file.close();
28             girl_file.close();
29             gift_file.close();
30         }catch(IOException e){
31 
32         }
33 
34 
35 }
```

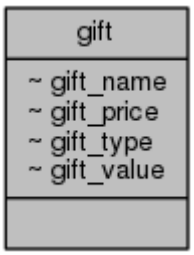
---

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

2 **csv_gen.java**

# gift

Collaboration diagram for gift:

IMAGE 


### Package Attributes

String **gift_name**
int **gift_price**
String **gift_type**
int **gift_value**

---

### Detailed Description

---

### Member Data Documentation

**String gift.gift_name[package]**

**int gift.gift_price[package]**

**String gift.gift_type[package]**

**int gift.gift_value[package]**

---

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

3 **gift.java**

# Girls

Collaboration diagram for Girls:


IMAGE

### Package Functions

void **happiness** (**gift** gf[], int n)
boolean **is_eligible** (int budget)

### Package Attributes

int **attar**
String **boyf**
int **expense**
String **girl_type**
double **happiness_girl**
int **intelligence**
String **name**
String **status**

---

### Detailed Description

---

### Member Function Documentation

### void Girls.happiness (gift *gf*[], int *n*)[inline], [package]

```
   21                                {
   22
   23        String type = this.girl_type;
   24        int i;
   25        if(type.equals("Choosy")){
   26                                int sum = 0;
   27                                for(i=0;i<n;i++){
   28                                        sum += gf[i].gift_price;
   29                                        if(gf[i].gift_type.equals("Luxury"))
{
   30                                                sum += 2*(gf[i].gift_value);
   31                                            //  System.out.println(sum);
   32                                        }
   33                                }
   34                                if(sum > this.expense){
   35                                    this.happiness_girl = Math.log(sum -
this.expense);
   36                                }
   37                                else{
   38                                    this.happiness_girl = 0;
   39                                }
   40
```

```
41         }else if(type.equals("Normal")){
42
43                                 int sum = 0;
44                                 for(i=0;i<n;i++){
45                                         sum = sum
+gf[i].gift_price+gf[i].gift_value;
46                                 }
47                                 if(sum > this.expense){
48                                     this.happiness_girl = sum -
this.expense;
49                                 }
50                                 else{
51                                     this.happiness_girl = 0;
52                                 }
53
54
55         }else if(type.equals("Desperate")){
56                                 int sum = 0;
57                                 for(i=0;i<n;i++){
58                                         sum +=gf[i].gift_price;
59                                 }
60                                 if(sum > this.expense){
61                                     this.happiness_girl = Math.exp(sum -
this.expense);
62                                 }
63                                 else{
64                                     this.happiness_girl = 0;
65                                 }
66
67
68             }
69     }
```

**boolean Girls.is_eligible (int *budget*)[inline], [package]**

```
12                                    {
13     if (this.expense <= budget) {
14             return true;
15         }
16     else {
17             return false;
18         }
19     }
```

---

**Member Data Documentation**

**int Girls.attar[package]**

**String Girls.boyf[package]**

**int Girls.expense[package]**

**String Girls.girl_type[package]**

**double Girls.happiness_girl[package]**

**int Girls.intelligence[package]**

**String Girls.name[package]**

**String Girls.status[package]**

---

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

4 **Girls.java**

# MyLogger

Collaboration diagram for MyLogger:

IMAGE 


### Static Package Functions

static void **logfile** (String str)

---

### Detailed Description

---

### Member Function Documentation

### static void MyLogger.logfile (String *str*)[inline], [static], [package]

```
 4                                       {
 5
 6
 7         try(FileWriter log = new FileWriter("log.txt")){
 8             log.write(str);
 9         }catch(IOException e){
10             e.printStackTrace();
11         }
12     }
```

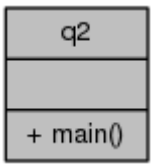
---

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

5 **MyLogger.java**

# q2

Collaboration diagram for q2:

IMAGE 


### Static Public Member Functions

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

---

### Detailed Description

---

### Member Function Documentation

#### static void q2.main (String *args*[]) throws IOException`[inline]`, `[static]`

```
 7                                                                {
 8         int i,j;
 9         i=0;
10         j=0;
11         int m=0;
12         Boys b[] = new Boys[1000];
13         csv_gen csv_files = new csv_gen();
14         csv_files.csv();
15         String csvFile = "boy.csv";
16         File out=new File("log.txt");
17         String line = "";
18         String csvSplit = ",";
19         BufferedReader buff = null;
20         try{
21             buff = new BufferedReader(new FileReader(csvFile));
22             while((line = buff.readLine() )!= null){
23                 String[] boys_table = line.split(csvSplit);
24
25                 b[i] = new Boys();
26                 b[i].name = boys_table[0];
27                 b[i].intelligence = Integer.parseInt(boys_table[1]);
28                 b[i].attractivness = Integer.parseInt(boys_table[2]);
29                 b[i].min_attr_req = Integer.parseInt(boys_table[3]);
30                 b[i].budget = Integer.parseInt(boys_table[4]);
31                 b[i].status = boys_table[5];
32                 b[i].boy_type= boys_table[6];
33                 b[i].girlf="";
34                 b[i].happiness_boy = 0;
35                 i++;
36             }
37         }
38
39         catch(FileNotFoundException e){
40             e.printStackTrace();
41         }catch(IOException e){
42             e.printStackTrace();
43         }
44
45         Girls g[] = new Girls[1000];
46         csvFile = "girl.csv";
47         line = "";
48         csvSplit = ",";
49         buff = null;
50
51
52         try{
```

```
            buff = new BufferedReader(new FileReader(csvFile));
            while((line = buff.readLine() )!= null){
                String[] girls_table = line.split(csvSplit);

                g[j] = new Girls();
                g[j].name = girls_table[0];
                g[j].attar = Integer.parseInt(girls_table[1]);
                g[j].expense = Integer.parseInt(girls_table[2]);
                g[j].intelligence = Integer.parseInt(girls_table[3]);
                g[j].status = girls_table[4];
                g[j].girl_type = girls_table[5];
                g[j].boyf = "";
                g[j].happiness_girl= 0;

                j++;
            }
        }
        catch(FileNotFoundException e){
            e.printStackTrace();
        }catch(IOException e){
            e.printStackTrace();
        }finally{
                if(buff != null){
                    try{
                        buff.close();
                    }catch(IOException e){
                    e.printStackTrace();
                    }
                }
        }


        gift gf[] = new gift[1000];
        csvFile = "gift.csv";
        line = "";
        csvSplit = ",";
        buff = null;


        try{
            buff = new BufferedReader(new FileReader(csvFile));
            while((line = buff.readLine() )!= null){
                String[] gifts_table = line.split(csvSplit);
            //  System.out.println(boy[0] + boy[1] + boy[2] + boy [3]);
                gf[m] = new gift();
                gf[m].gift_name = gifts_table[0];
                gf[m].gift_price = Integer.parseInt(gifts_table[1]);

                gf[m].gift_value = Integer.parseInt(gifts_table[2]);
                gf[m].gift_type = gifts_table[3];

                m++;

            }
        }
        catch(FileNotFoundException e){
            e.printStackTrace();
        }catch(IOException e){
            e.printStackTrace();
        }finally{
                if(buff != null){
                    try{
                        buff.close();
                    }catch(IOException e){
                    e.printStackTrace();
                    }
                }
        }

    //  Arrays.sort( gft);


        gift tem;
        int m1,n;
        for(m1=0;m1<m;m1++){
            for(n=m1+1;n<m;n++){
                if(gf[m1].gift_price > gf[n].gift_price){
```

```
  130                     tem= gf[m1];
  131                     gf[m1]=gf[n];
  132                     gf[n]=tem;
  133                 }
  134             }
  135         }
  136 
  137 
  138 
  139         int count = 0;
  140         int c,d;
  141         Couple gbcouple[] = new Couple[100];
  142         for(c=0;c<i;c++){
  143             for(d=0;d<j;d++){
  144                 if(b[c].is_eligible(g[d].expense,g[d].attar) &&
b[c].status.equals("Single") && g[d].status.equals("Single")){
  145                     b[c].girlf = g[d].name;
  146                     g[d].boyf = b[c].name;
  147                     b[c].status = "Is_committed";
  148                     g[d].status = "Is_committed";
  149                     count++;
  150                     String df= new SimpleDateFormat("dd/MM/yy
HH:mm:ss").format(new Time(System.currentTimeMillis()));
  151                     String output =  df+" "+"Boy : "+b[c].name + " is
Committed with " +"Girl : "+g[d].name + "\n" ;
  152                     System.out.println(output);
  153                     try(BufferedWriter buffer=new BufferedWriter(new
FileWriter(out,true))){
  154                         buffer.write(output);
  155 
  156                     }
  157                     gbcouple[count-1] = new Couple(b[c],g[d]);
  158 
  159                 }
  160             }
  161 
  162 
  163 
  164 
  165 
  166         }
  167 
  168         for(n=0;n<count;n++){
  169             gbcouple[n].calc_happiness();
  170             gbcouple[n].calc_affinity();
  171             gbcouple[n].calc_gifts(gf,m);
  172 }
  173         Couple cu;
  174         int x,y;
  175         for(x=0;x<count;x++){
  176             for(y=x+1;y<count;y++){
  177                 if(gbcouple[x].happiness < gbcouple[y].happiness){
  178                     cu= gbcouple[x];
  179                     gbcouple[x]=gbcouple[y];
  180                     gbcouple[y]=cu;
  181                 }
  182             }
  183         }
  184         Random happi = new Random();
  185         int h = happi.nextInt(count) +1;
  186         System.out.println("Best " +h+ " happiest Couples :"+"\n");
  187         for(x=0;x<h;x++){
  188         System.out.println(gbcouple[x].boy.name+" and
"+gbcouple[x].girl.name+"\n");
  189 
  190         String happy = "happiness : "+ gbcouple[x].boy.name+" and
"+gbcouple[x].girl.name+" & happiness is "+gbcouple[x].happiness +"\n";
  191         try(BufferedWriter buffer=new BufferedWriter(new
FileWriter(out,true))){
  192             buffer.write(happy);
  193 
  194         }}
  195         for(x=0;x<count;x++){
  196             for(y=x+1;y<count;y++){
  197                 if(gbcouple[x].affinity < gbcouple[y].affinity){
  198                     cu= gbcouple[x];
  199                     gbcouple[x]=gbcouple[y];
```

```
  200                     gbcouple[y]=cu;
  201                 }
  202             }
  203         }
  204         System.out.println("Best " +h+ " Compaitable Couples :"+"\n");
  205 
  206         for(x=0;x<h;x++){
  207             System.out.println(gbcouple[x].boy.name+" and
"+gbcouple[x].girl.name+"\n");
  208 
  209         String Comp = "compaitable : "+ gbcouple[x].boy.name+" and
"+gbcouple[x].girl.name+" & compatibality is "+gbcouple[x].affinity +"\n";
  210         try(BufferedWriter buffer=new BufferedWriter(new
FileWriter(out,true))){
  211             buffer.write(Comp);
  212         }
  213         }
  214     }
```

---

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

6 **q2.java**

# File Documentation

## Boys.java File Reference

### Classes

class **Boys**