

Documentation

API Documentation

April 9, 2017

Contents

Contents	1
1 Module boy_geek	2
1.1 Variables	2
1.2 Class GeekBoy	2
1.2.1 Methods	2
2 Module boy_generous	3
2.1 Variables	3
2.2 Class GenerousBoy	3
2.2.1 Methods	3
3 Module boy_miser	4
3.1 Variables	4
3.2 Class MiserBoy	4
3.2.1 Methods	4
4 Module boys	5
4.1 Variables	5
4.2 Class Boy	5
4.2.1 Methods	5
5 Module couple	6
5.1 Variables	6
5.2 Class Couple	6
5.2.1 Methods	6
6 Module gift_essential	7
6.1 Variables	7
6.2 Class EssentialGift	7
6.2.1 Methods	7
7 Module gift_luxury	8
7.1 Variables	8
7.2 Class LuxuryGift	8
7.2.1 Methods	8
8 Module gift_utility	9

8.1	Variables	9
8.2	Class UtilityGift	9
8.2.1	Methods	9
9	Module gifts	10
9.1	Variables	10
9.2	Class Gift	10
9.2.1	Methods	10
10	Module girl_choosy	11
10.1	Variables	11
10.2	Class ChoosyGirl	11
10.2.1	Methods	11
11	Module girl_desperate	12
11.1	Variables	12
11.2	Class DesperateGirl	12
11.2.1	Methods	12
12	Module girl_normal	13
12.1	Variables	13
12.2	Class NormalGirl	13
12.2.1	Methods	13
13	Module girls	14
13.1	Variables	14
13.2	Class Girl	14
13.2.1	Methods	14
14	Module q6_driver	15
14.1	Functions	15
14.2	Variables	16
15	Module utility	17
15.1	Functions	17
15.2	Variables	17

1 Module boy_geek

1.1 Variables

Name	Description
<code>__package__</code>	Value: None

1.2 Class GeekBoy

boys.Boy —
 boy_geek.GeekBoy

boy class for geek boys

1.2.1 Methods

<code>__init__(self, name, atr, gfbudget, intelli, min_atr_req, type)</code>
constructor
Overrides: boys.Boy.__init__

Inherited from boys.Boy(Section 4.2)

is_elligible()

2 Module *boy_generous*

2.1 Variables

Name	Description
<code>__package__</code>	Value: None

2.2 Class *GenerousBoy*

```

boys.Boy └─
           boy_generous.GenerousBoy

```

boy class for generous boys

2.2.1 Methods

<code>__init__(self, name, atr, gfbudget, intelli, min_atr_req, type)</code>
constructor
Overrides: <code>boys.Boy.__init__</code>

Inherited from `boys.Boy`(Section 4.2)

`is_elligible()`

3 Module boy_miser

3.1 Variables

Name	Description
<code>__package__</code>	Value: None

3.2 Class MiserBoy

```

boys.Boy └─
           boy_miser.MiserBoy

```

boy class for miser boys

3.2.1 Methods

<code>__init__(self, name, atr, gfbudget, intelli, min_atr_req, type)</code>
constructor
Overrides: boys.Boy.__init__

Inherited from boys.Boy(Section 4.2)

`is_elligible()`

4 Module boys

4.1 Variables

Name	Description
<code>__package__</code>	Value: None

4.2 Class Boy

Known Subclasses: `boy_geek.GeekBoy`, `boy_generous.GenerousBoy`, `boy_miser.MiserBoy`
 boy class for all boys

4.2.1 Methods

<code>__init__(self, name, atr, gfbudget, intelli, min_atr_req, type)</code>
constructor

<code>is_elligible(self, mbudget, atr)</code>
checks the elligibility of a given Girl, for the current instance of Boy class

5 Module couple

5.1 Variables

Name	Description
<code>__package__</code>	Value: None

5.2 Class Couple

couple class for all couples

5.2.1 Methods

<code>__init__(self, boy, girl)</code>
constructor

<code>set_happiness(self)</code>
set the happiness of a couple

<code>set_compatibility(self)</code>
set the compatibility of a couple

6 Module `gift_essential`

6.1 Variables

Name	Description
<code>__package__</code>	Value: None

6.2 Class `EssentialGift`

`gifts.Gift` └─
 `gift_essential.EssentialGift`

gift class for essential gifts

6.2.1 Methods

<code>__init__</code> (<i>self, name, price, value, type</i>)
constructor
Overrides: <code>gifts.Gift.__init__</code>

7 Module `gift_luxury`

7.1 Variables

Name	Description
<code>__package__</code>	Value: None

7.2 Class `LuxuryGift`

`gifts.Gift` —
`gift_luxury.LuxuryGift`

gift class for luxury gifts

7.2.1 Methods

<code>__init__</code> (<i>self</i> , <i>name</i> , <i>price</i> , <i>value</i> , <i>type</i> , <i>lxry_rtnng</i> , <i>difficulty</i>)
constructor
Overrides: <code>gifts.Gift.__init__</code>

8 Module `gift_utility`

8.1 Variables

Name	Description
<code>__package__</code>	Value: None

8.2 Class `UtilityGift`

`gifts.Gift` —
`gift_utility.UtilityGift`

gift class for utility gifts

8.2.1 Methods

<code>__init__</code> (<i>self</i> , <i>name</i> , <i>price</i> , <i>value</i> , <i>type</i> , <i>utlty_value</i> , <i>utlty_class</i>)
constructor
Overrides: <code>gifts.Gift.__init__</code>

9 Module gifts

9.1 Variables

Name	Description
<code>__package__</code>	Value: None

9.2 Class Gift

Known Subclasses: `gift_essential.EssentialGift`, `gift_luxury.LuxuryGift`, `gift_utility.UtilityGift`
 gift class for all gifts

9.2.1 Methods

<code>__init__</code> (<i>self</i> , <i>name</i> , <i>price</i> , <i>value</i> , <i>type</i>)
constructor

10 Module girl_choosy

10.1 Variables

Name	Description
<code>__package__</code>	Value: None

10.2 Class ChoosyGirl

girls.Girl —
 girl_choosy.ChoosyGirl

girl class for choosy girls

10.2.1 Methods

<code>__init__(self, name, atr, mbudget, intelli, type)</code>
constructor
Overrides: girls.Girl.__init__

Inherited from girls.Girl(Section 13.2)

is_elligible()

11 Module `girl_desperate`

11.1 Variables

Name	Description
<code>__package__</code>	Value: None

11.2 Class `DesperateGirl`

girls.Girl —
 girl_desperate.DesperateGirl

girl class for desperate girls

11.2.1 Methods

<code>__init__(self, name, atr, mbudget, intelli, type)</code>
constructor
Overrides: girls.Girl.__init__

Inherited from girls.Girl(Section 13.2)

`is_elligible()`

12 Module girl_normal

12.1 Variables

Name	Description
<code>__package__</code>	Value: None

12.2 Class NormalGirl

girls.Girl —
 girl_normal.NormalGirl

girl class for normal girls

12.2.1 Methods

<code>__init__(self, name, atr, mbudget, intelli, type)</code>
constructor
Overrides: girls.Girl. <code>__init__</code>

Inherited from girls.Girl(Section 13.2)

`is_elligible()`

13 Module girls

13.1 Variables

Name	Description
<code>__package__</code>	Value: None

13.2 Class Girl

Known Subclasses: `girl_choosy.ChoosyGirl`, `girl_desperate.DesperateGirl`, `girl_normal.NormalGirl`
 girl class for all girls

13.2.1 Methods

<code>__init__(self, name, atr, mbudget, intelli, type)</code>
constructor

<code>is_elligible(self, gfbudget)</code>
checks the elligibility of a given Boy, for the current instance of Girl class

14 Module q6_driver

14.1 Functions

allocate()

reads and stores the input from the boys.csv and girls.csv files and then makes the valid couples

calculate_happiness(B, G, C)

reads and stores the inputs from the gifts.csv file and provide gift exchanges between the couples

set_girl_happiness($c, v1, v2$)

sets the happiness of a girl according to her type

hp_miser(GFT, c)

provides gifting logic for Miser type Boys and sets the Happiness of the committed Boy and the whole couple, also sets the Compatibility of the couple

hp_generous(GFT, c)

provides gifting logic for Generous type Boys and sets the Happiness of the committed Boy and the whole couple, also sets the Compatibility of the couple

hp_geek(GFT, c)

provides gifting logic for Geek type Boys and sets the Happiness of the committed Boy and the whole couple, also sets the Compatibility of the couple

add_bf(c, BB)

Adds the ex-boyfriends of a girl in a list so that, the girl is not allotted any of the ex-boyfriends

check_in_list(r, BB)

Checks if a given boy (i.e r.boy) is an ex-boyfriend of the given girl (i.e r.girl) or not

loop_val(B, G, GFT, C)

Valentines day becomes t days long

print_lh (C, t, k)

prints the Couples having happiness less than t

newallocate (B, G, C, GFT, BB, k)
--

allocates new boys to the girls who broke up
--

new_gifting (B, G, C, GFT, NC)

gifting happens for the newly formed (after break-up) couples

14.2 Variables

Name	Description
__package__	Value: None

15 Module utility

15.1 Functions

utility()
creates the input csv files

create (<i>file</i> , <i>list</i>)
writes to csv files

15.2 Variables

Name	Description
__package__	Value: None

Index

- boy_geek (*module*), 2
 - boy_geek.GeekBoy (*class*), 2
- boy_generous (*module*), 3
 - boy_generous.GenerousBoy (*class*), 3
- boy_miser (*module*), 4
 - boy_miser.MiserBoy (*class*), 4
- boys (*module*), 5
 - boys.Boy (*class*), 5
 - boys.Boy.__init__ (*method*), 5
 - boys.Boy.is_elligible (*method*), 5
- couple (*module*), 6
 - couple.Couple (*class*), 6
 - couple.Couple.__init__ (*method*), 6
 - couple.Couple.set_compatibility (*method*), 6
 - couple.Couple.set_happiness (*method*), 6
- gift_essential (*module*), 7
 - gift_essential.EssentialGift (*class*), 7
- gift_luxury (*module*), 8
 - gift_luxury.LuxuryGift (*class*), 8
- gift_utility (*module*), 9
 - gift_utility.UtilityGift (*class*), 9
- gifts (*module*), 10
 - gifts.Gift (*class*), 10
 - gifts.Gift.__init__ (*method*), 10
- girl_choosy (*module*), 11
 - girl_choosy.ChoosyGirl (*class*), 11
- girl_desperate (*module*), 12
 - girl_desperate.DesperateGirl (*class*), 12
- girl_normal (*module*), 13
 - girl_normal.NormalGirl (*class*), 13
- girls (*module*), 14
 - girls.Girl (*class*), 14
 - girls.Girl.__init__ (*method*), 14
 - girls.Girl.is_elligible (*method*), 14
- q6_driver (*module*), 15–16
 - q6_driver.add_bf (*function*), 15
 - q6_driver.allocate (*function*), 15
 - q6_driver.calculate_happiness (*function*), 15
 - q6_driver.check_in_list (*function*), 15
 - q6_driver.hp_geek (*function*), 15
 - q6_driver.hp_generous (*function*), 15
 - q6_driver.hp_miser (*function*), 15
 - q6_driver.loop_val (*function*), 15
 - q6_driver.new_gifting (*function*), 16
 - q6_driver.newallocate (*function*), 16
 - q6_driver.print_lh (*function*), 15
 - q6_driver.set_girl_happiness (*function*), 15
- utility (*module*), 17
 - utility.create (*function*), 17
 - utility.utility (*function*), 17