Documentation

API Documentation

April 9, 2017

Contents

1 Module boy_geek

1.1 Variables

Name	**Description**
__package__	**Value:** `None`

1.2 Class GeekBoy

boys.Boy ─┐
boy_geek.GeekBoy

boy class for geek boys

1.2.1 Methods

__init__(*self*, *name*, *atr*, *gfbudget*, *intelli*, *min_atr_req*, *type*)

constructor

Overrides: boys.Boy.__init__

Inherited from boys.Boy(Section 4.2)

is_elligible()

2 Module boy_generous

2.1 Variables

Name	Description
__package__	**Value:** `None`

2.2 Class GenerousBoy

```
boys.Boy ─┐
          │
          boy_generous.GenerousBoy
```

boy class for generous boys

2.2.1 Methods

__init__(*self*, *name*, *atr*, *gfbudget*, *intelli*, *min_atr_req*, *type*)

constructor

Overrides: boys.Boy.__init__

Inherited from boys.Boy(Section 4.2)

is_elligible()

3 Module boy_miser

3.1 Variables

Name	Description
__package__	**Value:** `None`

3.2 Class MiserBoy

boys.Boy ┐
    boy_miser.MiserBoy

boy class for miser boys

3.2.1 Methods

__init__(*self*, *name*, *atr*, *gfbudget*, *intelli*, *min_atr_req*, *type*)

constructor

Overrides: boys.Boy.__init__

Inherited from boys.Boy(Section 4.2)

is_elligible()

4 Module boys

4.1 Variables

Name	Description
__package__	**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

__init__(*self*, *name*, *atr*, *gfbudget*, *intelli*, *min_atr_req*, *type*)

constructor

is_elligible(*self*, *mbudget*, *atr*)

checks the elligibility of a given Girl, for the current instance of Boy class

5 Module couple

5.1 Variables

Name	**Description**
__package__	**Value:** `None`

5.2 Class Couple

couple class for all couples

5.2.1 Methods

__init__(*self*, *boy*, *girl*)

constructor

set_happiness(*self*)

set the happiness of a couple

set_compatibility(*self*)

set the compatibility of a couple

6 Module gift_essential

6.1 Variables

Name	Description
__package__	**Value:** `None`

6.2 Class EssentialGift

```
gifts.Gift ─┐
            gift_essential.EssentialGift
```

gift class for essential gifts

6.2.1 Methods

__init__(*self*, *name*, *price*, *value*, *type*)

constructor

Overrides: gifts.Gift.__init__

7 Module gift_luxury

7.1 Variables

Name	Description
__package__	**Value:** `None`

7.2 Class LuxuryGift

gifts.Gift ─┐
gift_luxury.LuxuryGift

gift class for luxury gifts

7.2.1 Methods

__init__(*self*, *name*, *price*, *value*, *type*, *lxry_rtng*, *difficulty*)

constructor

Overrides: gifts.Gift.__init__

8 Module gift_utility

8.1 Variables

Name	Description
___package___	**Value:** `None`

8.2 Class UtilityGift

gifts.Gift ─┐
gift_utility.UtilityGift

gift class for utility gifts

8.2.1 Methods

___init___(*self*, *name*, *price*, *value*, *type*, *utlty_value*, *utlty_class*)

constructor

Overrides: gifts.Gift.___init___

9 Module gifts

9.1 Variables

Name	Description
__package__	**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

__init__(*self*, *name*, *price*, *value*, *type*)

constructor

10 Module girl_choosy

10.1 Variables

Name	Description
__package__	**Value:** `None`

10.2 Class ChoosyGirl

```
girls.Girl ─┐
            girl_choosy.ChoosyGirl
```

girl class for choosy girls

10.2.1 Methods

__init__(*self*, *name*, *atr*, *mbudget*, *intelli*, *type*)

constructor

Overrides: girls.Girl.__init__

Inherited from girls.Girl(Section 13.2)

is_elligible()

11 Module girl_desperate

11.1 Variables

Name	Description
__package__	**Value:** `None`

11.2 Class DesperateGirl

girls.Girl ─┐
girl_desperate.DesperateGirl

girl class for desperate girls

11.2.1 Methods

__init__(*self*, *name*, *atr*, *mbudget*, *intelli*, *type*)

constructor

Overrides: girls.Girl.__init__

Inherited from girls.Girl(Section 13.2)

is_elligible()

12 Module girl_normal

12.1 Variables

Name	Description
__package__	**Value:** `None`

12.2 Class NormalGirl

girls.Girl ─┐
girl_normal.NormalGirl

girl class for normal girls

12.2.1 Methods

__init__(*self*, *name*, *atr*, *mbudget*, *intelli*, *type*)

constructor

Overrides: girls.Girl.__init__

Inherited from girls.Girl(Section 13.2)

is_elligible()

13 Module girls

13.1 Variables

Name	Description
__package__	**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

__init__(*self*, *name*, *atr*, *mbudget*, *intelli*, *type*)

constructor

is_elligible(*self*, *gfbudget*)

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 commited 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 commited 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 commited 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 alloted 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(*file*, *list*)

writes to csv files

15.2 Variables

Name	Description
__package__	**Value:** `None`

Index