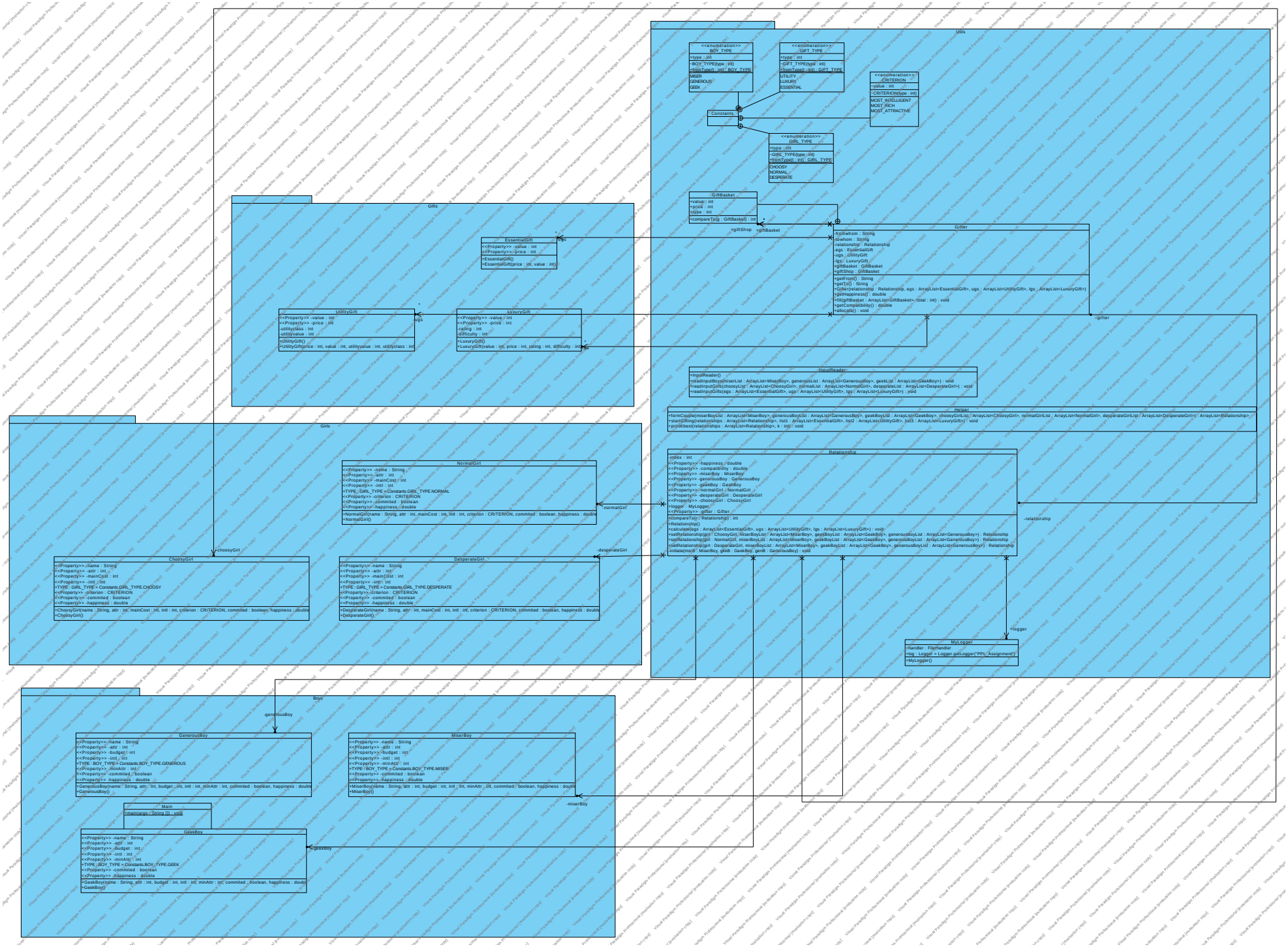
Utils
<<enumeration>>
BOY_TYPE
+type : int
-BOY_TYPE(type : int)
+fromType(t : int) : BOY_TYPE
MISER
GENEROUS
GEEK
<<enumeration>>
GIFT_TYPE
+type : int
-GIFT_TYPE(type : int)
+fromType(t : int) : GIFT_TYPE
UTILITY
LUXURY
ESSENTIAL
<<enumeration>>
CRITERION
-value : int
-CRITERION(type : int)
MOST_INTELLIGENT
MOST_RICH
MOST_ATTRACTIVE
Constants
<<enumeration>>
GIRL_TYPE
+type : int
-GIRL_TYPE(type : int)
+fromType(t : int) : GIRL_TYPE
CHOOSY
NORMAL
DESPERATE
GiftBasket
+value : int
+price : int
+type : int
+compareTo(g : GiftBasket) : int
Gifts
EssentialGift
<<Property>> -value : int
<<Property>> -price : int
-utilityclass : int
-utilityvalue : int
+UtilityGift()
+UtilityGift(price : int, value : int, utilityvalue : int, utilityclass : int)
UtilityGift
<<Property>> -value : int
<<Property>> -price : int
-rating : int
-difficulty : int
+LuxuryGift()
+LuxuryGift(value : int, price : int, rating : int, difficulty : int)
LuxuryGift
Gifter
+homewhom : String
-towhom : String
-relationship : Relationship
-egs : EssentialGift
-ugs : UtilityGift
-lgs : LuxuryGift
-giftBasket : GiftBasket
-giftShop : GiftBasket
+getFrom() : String
+getTo() : String
+Gifter(relationship : Relationship, egs : ArrayList<EssentialGift>, ugs : ArrayList<UtilityGift>, lgs : ArrayList<LuxuryGift>)
+getHappiness() : double
+fillGiftBasket : ArrayList<GiftBasket>, total : int) : void
+getCompatibility() : double
+allocate() : void
InputReader
+InputReader()
+readInputBoys(miserList : ArrayList<MiserBoy>, generousList : ArrayList<GenerousBoy>, geekList : ArrayList<GeekBoy>) : void
+readInputGirls(choosyList : ArrayList<ChoosyGirl>, normalList : ArrayList<NormalGirl>, desperateList : ArrayList<DesperateGirl>) : void
+readInputGifts(egs : ArrayList<EssentialGift>, ugs : ArrayList<UtilityGift>, lgs : ArrayList<LuxuryGift>) : void
Helper
+formCouple(miserBoyList : ArrayList<MiserBoy>, generousBoyList : ArrayList<GenerousBoy>, geekBoyList : ArrayList<GeekBoy>, choosyGirlList : ArrayList<ChoosyGirl>, normalGirlList : ArrayList<NormalGirl>, desperateGirlList : ArrayList<DesperateGirl>) : ArrayList<Relationship>
+startGifting(relationships : ArrayList<Relationship>, list1 : ArrayList<EssentialGift>, list2 : ArrayList<UtilityGift>, list3 : ArrayList<LuxuryGift>) : void
+printKbest(relationships : ArrayList<Relationship>, k : int) : void
Relationship
-index : int
<<Property>> -happiness : double
<<Property>> -compatibility : double
<<Property>> -miserBoy : MiserBoy
<<Property>> -generousBoy : GenerousBoy
<<Property>> -geekBoy : GeekBoy
<<Property>> -normalGirl : NormalGirl
<<Property>> -desperateGirl : DesperateGirl
<<Property>> -choosyGirl : ChoosyGirl
+logger : MyLogger
<<Property>> -gifter : Gifter
+compareTo() : Relationship) : int
+Relationship()
+calculate(egs : ArrayList<EssentialGift>, ugs : ArrayList<UtilityGift>, lgs : ArrayList<LuxuryGift>) : void
+setRelationship(girl : ChoosyGirl, miserBoyList : ArrayList<MiserBoy>, geekBoyList : ArrayList<GeekBoy>, generousBoyList : ArrayList<GenerousBoy>) : Relationship
+setRelationship(girl : NormalGirl, miserBoyList : ArrayList<MiserBoy>, geekBoyList : ArrayList<GeekBoy>, generousBoyList : ArrayList<GenerousBoy>) : Relationship
+setRelationship(girl : DesperateGirl, miserBoyList : ArrayList<MiserBoy>, geekBoyList : ArrayList<GeekBoy>, generousBoyList : ArrayList<GenerousBoy>) : Relationship
+initate(mcB : MiserBoy, gkB : GeekBoy, genB : GenerousBoy) : void
MyLogger
-handler : FileHandler
+log : Logger = Logger.getLogger("PPL_Assignment")
+MyLogger()
Girls
NormalGirl
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -mainCost : int
<<Property>> -intl : int
+TYPE : GIRL_TYPE = Constants.GIRL_TYPE.NORMAL
<<Property>> -criterion : CRITERION
<<Property>> -commited : boolean
<<Property>> -happiness : double
+NormalGirl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+NormalGirl()
ChoosyGirl
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -mainCost : int
<<Property>> -intl : int
+TYPE : GIRL_TYPE = Constants.GIRL_TYPE.CHOOSY
<<Property>> -criterion : CRITERION
<<Property>> -commited : boolean
<<Property>> -happiness : double
+ChoosyGirl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+ChoosyGirl()
DesperateGirl
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -mainCost : int
<<Property>> -intl : int
+TYPE : GIRL_TYPE = Constants.GIRL_TYPE.DESPERATE
<<Property>> -criterion : CRITERION
<<Property>> -commited : boolean
<<Property>> -happiness : double
+DesperateGirl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+DesperateGirl()
Boys
GenerousBoy
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -budget : int
<<Property>> -intl : int
+TYPE : BOY_TYPE = Constants.BOY_TYPE.GENEROUS
<<Property>> -minAttr : int
<<Property>> -commited : boolean
<<Property>> -happiness : double
+GenerousBoy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+GenerousBoy()
MiserBoy
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -budget : int
<<Property>> -intl : int
<<Property>> -minAttr : int
+TYPE : BOY_TYPE = Constants.BOY_TYPE.MISER
<<Property>> -commited : boolean
<<Property>> -happiness : double
+MiserBoy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+MiserBoy()
Main
+main(args : String[]) : void
GeekBoy
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -budget : int
<<Property>> -intl : int
<<Property>> -minAttr : int
+TYPE : BOY_TYPE = Constants.BOY_TYPE.GEEK
<<Property>> -commited : boolean
<<Property>> -happiness : double
+GeekBoy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+GeekBoy()
-generousBoy
-geekBoy
-miserBoy
-choosyGirl
-desperateGirl
-normalGirl
-relationship
-gifter
+logger
-giftShop
+giftBasket
-egs
-ugs
-lgs