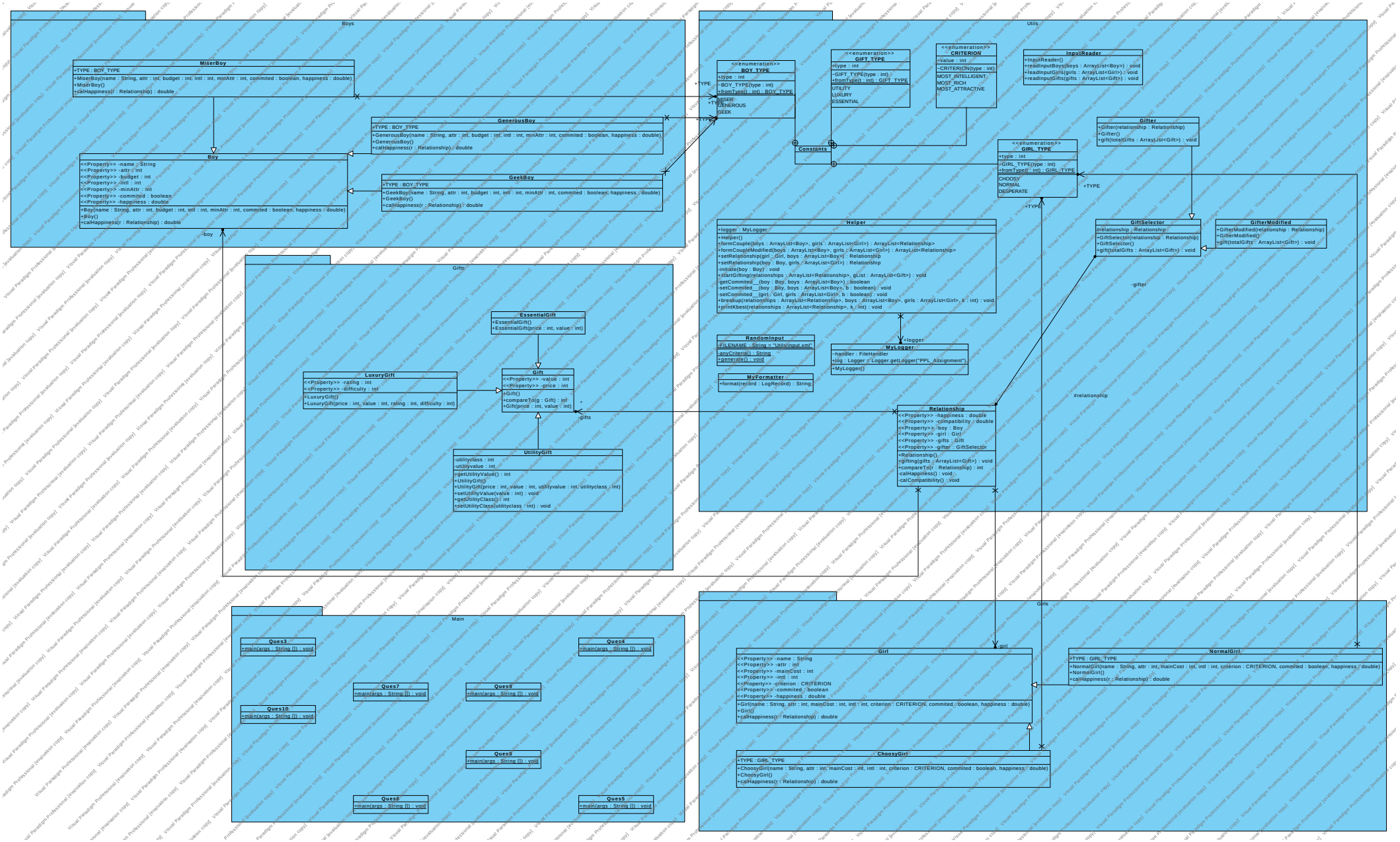
Boys
MiserBoy
+TYPE : BOY_TYPE
+Miser(Boy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+MiserBoy()
+calHappiness(r : Relationship) : double
Boy
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -budget : int
<<Property>> -intl : int
<<Property>> -minAttr : int
<<Property>> -commited : boolean
<<Property>> -happiness : double
+Boy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+Boy()
+calHappiness(r : Relationship) : double
GenerousBoy
+TYPE : BOY_TYPE
+GenerousBoy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+GenerousBoy()
+calHappiness(r : Relationship) : double
GeekBoy
+TYPE : BOY_TYPE
+GeekBoy(name : String, attr : int, budget : int, intl : int, minAttr : int, commited : boolean, happiness : double)
+GeekBoy()
+calHappiness(r : Relationship) : double
+TYPE
+TYPE
-boy
Utils
<<enumeration>>
BOY_TYPE
+type : int
-BOY_TYPE(type : int)
+fromType(t : int) : BOY_TYPE
MISER
GENEROUS
GEEK
Constants
<<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
InputReader
+InputReader()
+readInputBoys(boys : ArrayList<Boy>) : void
+readInputGirls(girls : ArrayList<Girl>) : void
+readInputGifts(gifts : ArrayList<Gift>) : void
Gifter
+Gifter(relationship : Relationship)
+Gifter()
+gift(totalGifts : ArrayList<Gift>) : void
<<enumeration>>
GIRL_TYPE
+type : int
-GIRL_TYPE(type : int)
+fromType(t : int) : GIRL_TYPE
CHOOSY
NORMAL
DESPERATE
+TYPE
+TYPE
Helper
+logger : MyLogger
+Helper()
+formCouple(boys : ArrayList<Boy>, girls : ArrayList<Girl>) : ArrayList<Relationship>
+formCoupleModified(boys : ArrayList<Boy>, girls : ArrayList<Girl>) : ArrayList<Relationship>
+setRelationship(girl : Girl, boys : ArrayList<Boy>) : Relationship
+setRelationship(boy : Boy, girls : ArrayList<Girl>) : Relationship
-initiate(boy : Boy) : void
+startGifting(relationships : ArrayList<Relationship>, gList : ArrayList<Gift>) : void
-getCommited__(boy : Boy, boys : ArrayList<Boy>) : boolean
-setCommited__(boy : Boy, boys : ArrayList<Boy>, b : boolean) : void
-setCommited__(girl : Girl, girls : ArrayList<Girl>, b : boolean) : void
+breakup(relationships : ArrayList<Relationship>, boys : ArrayList<Boy>, girls : ArrayList<Girl>, k : int) : void
+printNbest(relationships : ArrayList<Relationship>, k : int) : void
GiftSelector
#relationship : Relationship
+GiftSelector(relationship : Relationship)
+GiftSelector()
+gift(totalGifts : ArrayList<Gift>) : void
GifterModified
+GifterModified(relationship : Relationship)
+GifterModified()
+gift(totalGifts : ArrayList<Gift>) : void
-gifter
RandomInput
-FILENAME : String = "Utils/input.xml"
-anyCriteria() : String
+generate() : void
MyFormatter
+format(record : LogRecord) : String
+logger
MyLogger
-handler : FileHandler
+log : Logger = Logger.getLogger("PPL_Assignment")
+MyLogger()
#relationship
Relationship
<<Property>> -happiness : double
<<Property>> -compatibility : double
<<Property>> -boy : Boy
<<Property>> -girl : Girl
<<Property>> -gifts : Gift
<<Property>> -gifter : GiftSelector
+Relationship()
+gifting(gifts : ArrayList<Gift>) : void
+compareTo(r : Relationship) : int
-calHappiness() : void
-calCompatibility() : void
Gifts
EssentialGift
+EssentialGift()
+EssentialGift(price : int, value : int)
LuxuryGift
<<Property>> -rating : int
<<Property>> -difficulty : int
+LuxuryGift()
+LuxuryGift(price : int, value : int, rating : int, difficulty : int)
Gift
<<Property>> -value : int
<<Property>> -price : int
+Gift()
+compareTo(g : Gift) : int
+Gift(price : int, value : int)
*
-gifts
UtilityGift
-utilityclass : int
-utilityvalue : int
+getUtilityValue() : int
+UtilityGift()
+UtilityGift(price : int, value : int, utilityvalue : int, utilityclass : int)
+setUtilityValue(value : int) : void
+getUtilityClass() : int
+setUtilityClass(utilityclass : int) : void
Main
Ques3
+main(args : String []) : void
Ques4
+main(args : String []) : void
Ques7
+main(args : String []) : void
Ques8
+main(args : String []) : void
Ques10
+main(args : String []) : void
Ques9
+main(args : String []) : void
Ques6
+main(args : String []) : void
Ques5
+main(args : String []) : void
Girls
Girl
<<Property>> -name : String
<<Property>> -attr : int
<<Property>> -mainCost : int
<<Property>> -intl : int
<<Property>> -criterion : CRITERION
<<Property>> -commited : boolean
<<Property>> -happiness : double
+Girl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+Girl()
+calHappiness(r : Relationship) : double
-girl
NormalGirl
+TYPE : GIRL_TYPE
+NormalGirl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+NormalGirl()
+calHappiness(r : Relationship) : double
ChoosyGirl
+TYPE : GIRL_TYPE
+ChoosyGirl(name : String, attr : int, mainCost : int, intl : int, criterion : CRITERION, commited : boolean, happiness : double)
+ChoosyGirl()
+calHappiness(r : Relationship) : double