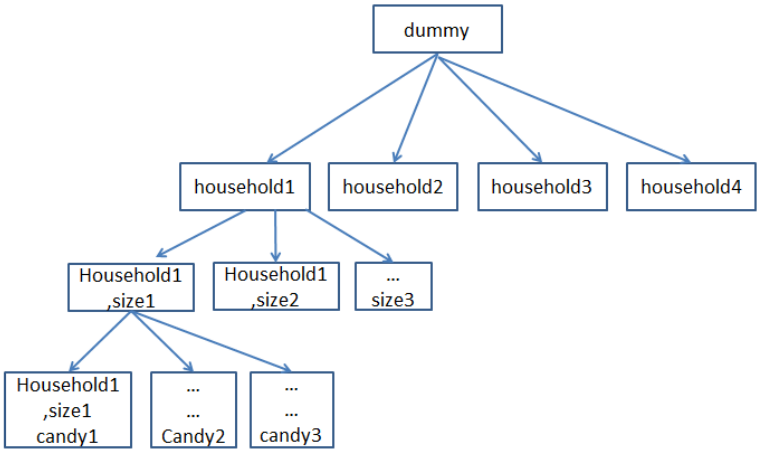
Assignment4- writeup

UML explaination

In this assignment, I create abstract class Household and its subclass- 4 types of households as well as abstract class Candy and its subclass- 4 sizes of candies. The reason why I choose to create 4 classes of candies of different size rather than 10 classes of candies of different types includes two aspects. First, in a specific house, there is always one size of candy contains different kinds of candy. I would like to use up to down method to search the candy, and the order is shown in the graph below. The logic is that when searching for a certain type of candy, it first searches for the house, then search the candy size in that house, finally determine whether this house has specific kind of candy. Second, the number of candy size is less than the number of candy type.



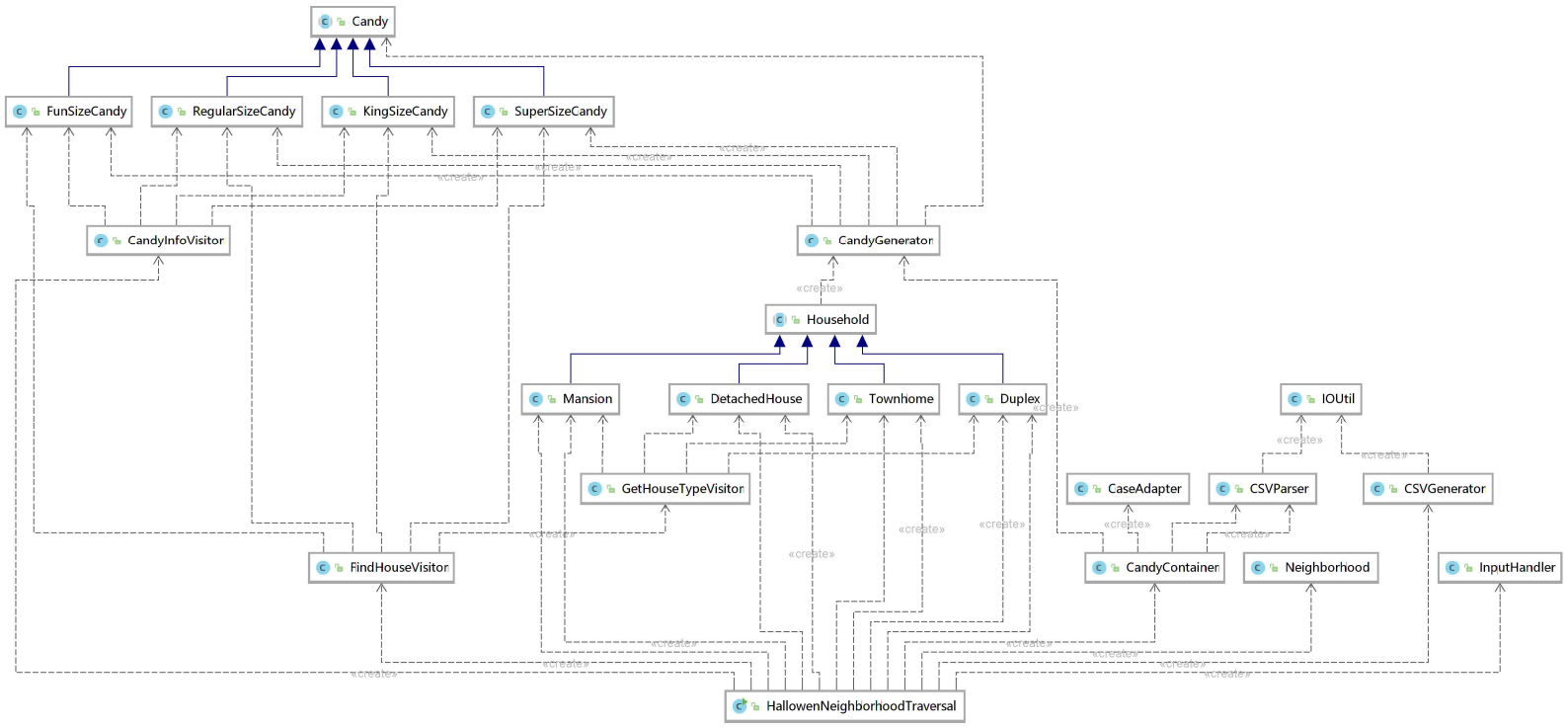
For the dummy node shown above, I use Neighborhood class to store the knowledge information. Whenever we create FindHouseVisitor, we need to put a Neighborhood with knowledge in. And it will search in each house one by one. The Neighborhood has hasNextHouse() and nextHouse() methods to traverse all the households.

Handle Exceptions

while handling input files using for loop, I catch the Exception when dealing with one file, and print the corresponding message. Then it will continue the loop and keep on handling the next file.

Design Pattern

In this assignment, I used Visitor Pattern because there are many classes with similar functions like candies of different size and different households. Using this pattern can make us easily add new functions to each similar classes in the future.


Candy
FunSizeCandy
RegularSizeCandy
KingSizeCandy
SuperSizeCandy
CandyInfoVisitor
CandyGenerator
Household
Mansion
DetachedHouse
Townhome
Duplex
GetHouseTypeVisitor
FindHouseVisitor
IOUtil
CaseAdapter
CSVParser
CSVGenerator
CandyContainer
Neighborhood
InputHandler
HallowenNeighborhoodTraversal
«create»