Task 1 pseudo code:

MAIN FUNCTIONS ONLY

Swap:

If biggest shopping list for a equal to A

Set variable to A

If biggest shopping list for a equal to B

Set variable to B

If biggest shopping list for a equal to C

Set variable to C

For each category

For each item in category

If the old item is an item in that category

If the old item is in the no substatutes category

Return old item

For each item in a category

Store new items stores in variable

If item in category is not old item and can be bought from shop with most items

Return new item

Substitutions(houseCount):

Find the store in which house requires most items for and store in variable.

Find the store in which house requires least items for and store in variable.

If the size of a is the smallest

Set a as smallest shop

Else if B is the smallest

Set B as smallest shop

Else if c is the smallest

Set C as smallest shop

If the size of A is the biggest

Set A as biggest shop

Else if B is the biggest

Set B as biggest shop

Else if C is the biggest

Set C as biggest shop

If A has fewest amount of items

For all items in shop A the house requires

Get all items alternatives using swap function and store in list

For all items in new items list

Get the new items number store in variable.

If new item in shop B

Append item to shop B shopping list for house.

Append item quantity to shop B shopping list for house.

If new item in shop C

Append item to shop C shopping list for house.

Append item quantity to shop C shopping list for house.

Clear items in house shopping list A

Clear item A quantities

If B has fewest amount of items

For all items in B the house requires

Get all items alternatives using swap function and store in list

For all items in new items list

Get the new items number store in variable.

If new item in shop A

Append item to shop A shopping list for house.

Append item quantity to shop A shopping list for house.

If new item in shop C

Append item to shop C shopping list for house.

Append item quantity to shop C shopping list for house.

Clear items in house shopping list A

Clear item A quantities

If C has fewest amount of items

For all items in C the house requires

Get all items alternatives using swap function and store in list

For all items in new items list

Get the new items number store in variable.

If new item in shop A

Append item to shop A shopping list for house.

Append item quantity to shop A shopping list for house.

If new item in shop B

Append item to shop B shopping list for house.

Append item quantity to shop B shopping list for house.

Clear items in house shopping list C

Clear item C quantities

**RecalculateMinShops()**

For each house

Define MinimumStoresList

If length of items required from shop A is greater than 0

Append shop A to MinimumStoresList

If length of items required from shop B is greater than 0

Append shop B to MinimumStoresList

If length of items required from shop C is greater than 0

Append shop C to MinimumStoresList

**MatchCombinations(combination, minimumCombos):**

For each combination set

Define match

For each shop in combination set

If minimumCombos equals a shop from combination

Increment match variable

If match equals the length of current combination of shops

Return true

**commonShopCombinations(getWeek):**

define combinations

for each house

if house is in week one

get minimumCombinations store in variable minimumCombos

if the length of combinations is zero

append first houses combination

else

define matchCondition set to false

matchCondition equal to matchCombinations() parse in combinations, minimumCombos

if matchCondition equals true

iterate loop one time

if matchCondition equals false

append minimumCombo to combinations.

**mergeShopLists(week, weekCombinations):**

for all weekCombinations

for all shops in weekCombinations

append shop to listOfShops

return listOfShops

**shoppingSceduleAddShops(Weeks, listofshops):**

for each in listofshops

for each in shoppingSceduleObjects

if current shoppingSceduleObject’s week is equal too week 1

if shoppingSceduleObject has no shop to buy from

set shop from listofShops current shop

break

**shoppingSceduleShoppingSort(week):**

for each house

if that houses week equals week

for each shoppingSceduleObject

if current shoppingSceduleObject