Final Project Diagram

Tuesday, November 20, 2018 10:23 AM

instance variables:

Scanner:	scan	reads from keyboard.
String:	I) username II) sure III) food	I) personalizes the experience (i.e. "welcome + username") II) scanner for "are you sure?" questions III) scanner for product input
Int	action	choose from a menu (switch)
Double	I) calories II) read	I) total calories II) scans limit values
Double[][]	Consumption [8][2]	stores limits for each category and current consumption [category][0=limits; 1=current consumption]
Boolean	exit	while false loops, when true exits the app
String[][]	I) fruits [13][5] II) dairy [7][5] III) legumes [7][5] IV) sugars [4][5] V) animals [8][5] VI) vegetables [18][5] VII) oils [14][5] VIII) cereals [18][5]	category[no of products][0=product name 1=portion (String) 2=calories per portion 3=measurement units 4=portion (Double)]

main

1. runs scanfiles() method
2. asks for username
3. prints main menu, where user can choose between 7 different actions
4. action #7 exits the app

exitapp()

1. prints summary into .txt file
2. exits the app

variables:

String	summaryfile	creates summary file
PrintWriter	theSummary	writes on summary file

seemanual()

prints the user manual

variables:

Scanner	manualfile	reads the user manual from a .txt file

limits()

lets the user enter custom limits, entering a negative value will leave the current limit as it is.

register()

1. asks the user for the product name
2. finds product in category array
3. asks user to enter portions or quantity
4. deduces the calories and portions and adds them to the total count
5. handles errors (i.e. not on menu, input mismatch, etc.)

variables:

String	I) food II) units	I) reads product name II) portion or units choice
Int	I) countb II) category III) product	I) step 2 counter II) category found on step 2 III) product found on step 2
Double	I) port II) quantity III) portions IV) cals	I) portion read from file converted from string to double II) reads the quantity consumed III) quantity/one portion (gets consumed portions) IV) product calories read from file converted from string to

	V) caloriesum	double V) multiplies calories by the consumed portions
Boolean	I) exitswitch II) exitinvalid	I) while true loops, when false exits switch cases II) while true loops, when false exits quantity or portions choice

summary()
prints current stats

reset()

1. asks user for validation
2. sets current consumption and total calories to 0

variables:

Int	countb	consumption reset counter
Boolean	exitReset	loops until user inputs a valid option

setLim()

1. lets the user choose between resetting the limits, setting new limits or leave them as they are
2. runs either option depending of the user's choice

variables:

Scanner	limitscan	re-scans the limits from the .txt file
Int	countb	reset limits counter
Boolean	I) exit2 II) exitsure	I) loops until the user inputs a valid option II) loops until the user decides whether to reset the limits or not

scanFiles()
scans the data from all .txt files, except for the summary file

variables:

Scanner	I) fruitscan II) dairyscan	Scanners for each file

	III) legumescan IV) sugarscan V) animalscan VI) vegetablescan VII) oilscan VIII) cerealscan IX) limitscan	
Int	I) counta II) countb III) countc	counters to loop for multidimensional arrays

printMenu()
prints the user menu

variables:

Int	countc=0	loops to print product, portion and calories

Files

- fruits.txt, dairy.txt, legumes.txt, etc. contain the information for each category, arranged as: product name, how much is one portion, calories, units and portion number
- limits contains the portion limits for each category
- summary is where the summary is printed when you exit the app
- manual contains the user manual