Assignment

Algorithm, Flowchart – Catering System

Atreya Bain, Chirag Bapat

Section: J1

Algorithm

Catering system transactions

Problem definition: Design and develop an integrated solution of a caterer billing system to run a small scale business in a day to day event transaction activities. The solution provides complete details of the valid business details with user friendly environment along with the report details.

Used variables and data structures:

Global variables:

1. cat\_details : Contains a list of miscellaneous caterer details. Consists of:
   1. name (String) : String to store name of Catering company
   2. taxp (Float) : Tax Percentage
2. menu : Containing the menu list. Consists of:
   1. num\_menu (Integer) : Number of items in the menu
   2. pieces (Array of custom structure) : Consists of number of each inventory item, which can be accessed by “pieces[i]” where i is the i-th element (Considering the array to be zero-indexed). Each of these items further consist of :-
      1. name (String) : Name of the food item
      2. sprice (Float) : Selling price of the item
      3. pcost (Float) : Production of the item
3. last\_invoice : Stores the most recently generated/used invoice in the program. As at most one invoice is required to be loaded at once, only one such structure is used.
   1. recep (String) : Contains the name of the recipient the invoice is addressed to.
   2. item\_numbers (Array of tuples): Each item in this tuple, represents the corresponding entry in the menu-list, and the corresponding quantity, where the former is stored, in the item’s 0th index.
   3. pieces\_len (Integer) : Contains the number of items in
4. Invoice\_list : Stores a list of names of generated bills to generate the reports from, and has:
   1. num\_invoice (Integer) : Number of invoices generated so far
   2. invoice\_name\_list (Array of strings) : Contain the list of names of invoices generated so far

Used subroutines

All variables have access to the above variables as they are global, and may or may not return a return a value to the main program, and subroutines returning a value maybe used to evaluate expressions, or can be stored into variables for future use.

The format for presenting a subroutine below is : <function\_name>( [Argument type]:[Arguments] )

1. read\_cat\_det() : Load the caterer details file if present and return 1, else return 0
2. write cat\_det() : Save the ‘cat\_details’ structure in a file for reference.
3. print\_cat\_det() : Print out the caterer details in a formatted manner
4. read\_menulist() : Load the menu list file if present and return 1, else return 0
5. write\_menulist() : Save the ‘menu’ structure in a file for use.
6. print\_menulist() : Print the menu list for reference
7. input\_item\_number() : Prints the menu, and returns the choice the user has submit
8. read\_invoice\_list() : Load from storage the list of invoices
9. write\_invoice() : Make an invoice, and save it to storage, and update the invoice list
10. read\_invoice() : Load an invoice from memory
11. print\_invoice() : Print the invoice that has been loaded onto the memory.
12. report(): Generate report of number of items sold, and total sales, profit and tax.
13. check\_substreq(String: string1, String: string2, Integer: position1, Integer: position2) : Checks if the strings are equal, between position1 and position 2 (including position1 and excluding position2)

Calls to the system to load a variable/structure state from storage has been denoted by the following functions:

1. FREAD( String: File name ) : Read a file name from memory
2. FWRITE( Variable/Data Structure, String: File name ) : Save to specified file path to storage
3. EXISTS(String: File name ) : Check if a given file exists or not, returns true (or 1) if exists, else returns false (or 0)
4. GET\_DATETIME() : Returns the current system date and time in a string format

Main Program:

Step 01: START

Step 02: flag=1

Step 03: flag\_hasloadedmenu = read\_menulist()

Step 04: flag\_hasloadedcat = read\_cat\_det()

Step 05: flag\_hasloadedinvoicelist = read\_invoice\_list()

Step 06: while ( flag )

Step 07: PRINT “”

Step 08: INPUT choice

Step 09: If ( choice = 1 ) then

Step 10: If ( flag\_hasloadedcat = 1 ) then

Step 11: write\_cat\_det()

Step 12: else

Step 13: PRINT “Caterer details not present”

Step 14: Elseif ( choice = 2 ) then

Step 15: flag\_hasloadedmenu = read\_menulist()

Step 16: print\_cat\_det()

Step 17: Elseif ( choice = 3 ) then

Step 18: if ( flag\_hasloadedmenu = 1 ) then

Step 19: write\_menulist()

Step 20: else

Step 21: PRINT “Menu List not present”

Step 22: Elseif ( choice = 4 ) then

Step 23: flag\_hasloadedmenu = read\_menulist()

Step 24: print\_menulist()

Step 25: Elseif ( choice = 5 ) then

Step 26: write\_invoice()

Step 27: Elseif ( choice = 6 ) then

Step 28: read\_invoice()

Step 29: print\_invoice()

Step 30: Elseif ( choice = 7 ) then

Step 31: report()

Step 32: Else

Step 33: PRINT “Invalid Choice”

Step 34: Endif

Step 35: End while

Step 36: STOP

Subroutines

1. read\_cat\_det() :

Step 01: START

Step 02: If( exists(“company.details”) ) then

Step 03: cat\_details = FREAD(“company.details”)

Step 04: RETURN 1

Step 05: Else

Step 06: RETURN 0

Step 07: Endif

1. write\_cat\_det()

Step 01: START

Step 02: PRINT “Enter company name”

Step 03: INPUT cat\_details.name

Step 04: PRINT “Enter Tax%”

Step 05: INPUT cat\_details.taxp

Step 06: FWRITE(cat\_details,“company.details”)

Step 07: RETURN

1. print\_cat\_det()

Step 01: START

Step 02: PRINT “Company Name: ”,cat\_details.name

Step 03: PRINT “Tax Percent: ” cat\_details.taxp

Step 04: RETURN

1. read\_menulist()

Step 01: START

Step 02: If( exists(“menu.details”) ) then

Step 03: menu = FREAD(“menu.details”)

Step 04: RETURN 1

Step 05: Else

Step 06: RETURN 0

Step 07: Endif

1. write\_menulist()

Step 01: START

Step 02: PRINT “Number of item in the menu:”

Step 03: INPUT menu.num\_item

Step 04: If (menu.num\_menu>128) then

Step 05: Goto Step 2

Step 06: for I = 0 to num\_item

Step 07: PRINT “Enter item name”

Step 08: INPUT menu.pieces[I].name

Step 09: PRINT “Enter item price and production cost”

Step 10: INPUT menu.pieces[I].sprice, menu.pieces[I].pcost

Step 11: end for

Step 12: FWRITE(menu,”menu.details”)

Step 13: RETURN

1. print\_menulist()

Step 01: START

Step 02: Print “Item Name Price Production”

Step 03: for I = 0 to menu.num\_menu

Step 04: PRINT menu.pieces[i].name, menu.pieces[i].sprice, menu.pieces[i].pcost

Step 05: end for

Step 06: RETURN

1. input\_item\_number()

Step 01: START

Step 02: print\_menulist()

Step 03: INPUT choice

Step 04: RETURN choice

1. read\_invoice\_list()

Step 01: START

Step 02: If( exists(“menu.details”) ) then

Step 03: menu = FREAD(“menu.details”)

Step 04: RETURN 1

Step 05: Else

Step 06: RETURN 0

Step 07: Endif

1. write\_invoice()

Step 01: START

Step 02: I = 0

Step 03: filename = GET\_DATETIME() + “.bill”

Step 04: PRINT “Invoice Recepient:”

Step 05: INPUT last\_invoice.rep

Step 06: PRINT “Enter item numbers

Step 07: while( ( buffer = input\_item\_number() ) ≠ 0 )

Step 08: last\_invoice.item\_numbers[I][0] = buffer-1

Step 09: PRINT “Enter item quantity”

Step 10: INPUT last\_invoice.item\_numbers[I][1]

Step 11: I = I + 1

Step 12: end while

Step 13: PRINT “Number of items ordered:”,I-1

Step 14: last\_invoice.pieces\_len = i-1

Step 15: if ( last\_invoice.pieces\_len =0 )

Step 16: RETURN 0;

Step 17: endif

Step 18: FWRITE(last\_invoice,filename)

Step 19: invoice\_list.invoice\_name\_list[invoice\_list.num\_invoice] = filename

Step 20: invoice\_list.num\_invoice = invoice\_list.num\_invoice + 1

Step 21: FWRITE(invoice\_list,”bill\_list.details”)

Step 22: RETURN last\_invoice.pieces\_len

1. read\_invoice()

Step 01: START

Step 02: if (!exists(“bill\_list.details”)) then

Step 03: invoice\_list.num\_invoice = 0

Step 04: else

Step 05: read\_invoice\_list

Step 06: endif

Step 07: if (invoice\_list.num\_invoice = 0) then

Step 08: PRINT “No invoices.”

Step 09: endif

Step 10: PRINT “Choose invoice to print”

Step 11: FOR I = 0 to invoice\_list.num\_invoice-1

Step 12: PRINT I+1,“ “,invoice\_list.invoice\_name\_list[I]

Step 13: end for

Step 14: INPUT CHOICE

Step 15: filename = invoice\_list.invoice\_name\_list[choice-1]

Step 16: last\_invoice = FREAD( filename )

Step 17: RETURN

1. print\_invoice()

Step 01: START

Step 02: PRINT last\_invoice.recep

Step 03: total\_price = 0

Step 04: PRINT “Items:”

Step 05: for I = 0 to last\_invoice.pieces\_len

Step 06: item\_number = last\_invoice.item\_numbers[i][0]

Step 07: item\_quantity = last\_invoice.item\_numbers[i][1]

Step 08: item\_name = menu.pieces[ item\_number ].name

Step 09: item\_price = menu.pieces[ item\_number ].sprice

Step 10: total\_price = total\_price + item\_price\*item\_quantity

Step 11: PRINT item\_name,item\_price,item\_quantity

Step 12: end for

Step 13: PRINT “Billed Amount”, total\_price

Step 14: RETURN

1. report()

Step 01: START

Step 02: fn = GET\_DATETIME() + “.bill”

Step 03: str\_datebounds = {0,4,6,8}

Step 04: num\_items = 0

Step 05: total\_sprice = 0

Step 06: total\_profit = 0

Step 07: total\_pcost = 0

Step 08: total\_tax = 0

Step 09: PRINT “Choose interval: 1. Year

2. Month

3. Day”

Step 10: INPUT interval

Step 11: for I = 0 to invoice\_list.num\_invoice

Step 12: check = check\_substreq(invoice\_list.invoice\_name\_list[i],fn,0,str\_datebounds[choice])

Step 13: if(check) then

Step 14: last\_invoice = FREAD( invoice\_list.invoice\_name\_list[I] )

Step 15: for J = 0 to last\_invoice.pieces\_len

Step 16: item\_number = last\_invoice.item\_numbers[i][0]

Step 17: item\_quantity = last\_invoice.item\_numbers[i][1]

Step 18: item\_number = last\_invoice.item\_numbers[i][0]

Step 19: item\_sprice = menu.pieces[ item\_number ].sprice

Step 20: item\_pcost = menu.pieces[ item\_number ].pcost

Step 21: total\_sprice = total\_sprice + item\_sprice \* item\_quantity

Step 22: total\_pcost = total\_pcost + item\_pcost \* item\_quantity

Step 23: num\_items = num\_items + item\_quantity

Step 24: end for

Step 25: end if

Step 26: end for

Step 27: if(num\_items ≠ 0) then

Step 28: total\_tax = total\_sprice \* cat\_details.taxp/100

Step 29: total\_profit = total\_sprice + total\_tax - total\_pcost

Step 30: PRINT “Totalling - ”

Step 31: PRINT “Number of items: ”,num\_items

Step 32: PRINT “Total tax: ”,total\_tax

Step 33: PRINT “Total Production Costs: ”,total\_pcost

Step 34: PRINT “Total Sale Price”,total\_sprice

Step 35: PRINT “Total Profit”,total\_profit

Step 36: else

Step 37: PRINT “No items”

Step 38: endif

Step 39: RETURN

1. check\_substreq(string1, string2, Integer: position1, position2)

Step 01: START

Step 02: fl = 1

Step 03: for I = position1 to position 2

Step 04: if(string[1] = string2[i])

Step 05: fl = 0

Step 06: GOTO STEP 09

Step 07: endif

Step 08: end for

Step 09: RETURN fl