UTAR
UNIVERSITI TUNKU ABDUL RAHMAN
Wholly owned by UTAR Education Foundation
(Co. No. 578227-M)
DU012(A)

UCCD 1004 PROGRAMMING CONCEPTS AND PRACTICES

ASSIGNMENT 2

Name:	Wan Yi Mun
Student ID:	20ACB03797
Programme:	Bachelor of Information System (Honours) Business Information System
Email:	yimun0304@1utar.my

Table of Contents

A. Objective

Since JKJY mini market want to computerize the cash registration process, so this system has been implemented and this system can allow JKJY mini market to view all the Item list that include item name, item code, price per Kg and purchasing weight for each item. Besides, it also allows JKJY mini market to add new Item into Item list and modify the existing items in item List. Furthermore, it also can allow JKJY mini market create and print the invoice. The invoice will include all the item name, price for one item which price per Kg multiple purchasing weight and grand total of all the item. Lastly, it will prompt the grand total of all items before closing the program.

B. Pseudocode

```
Begin
define SIZE equal to 100
main
End
```

main

```
Function main ( )
Begin
Declare goods_price[SIZE]
Declare goods_weight[SIZE]
Declare purchase_day
Declare purchase_date
Declare goods_name[SIZE]
Declare goods_code[SIZE]
Declare select
Declare cont
Declare number_goods equal to 0

Call readItemList

Do
    Display "1. Display Item List"
    Display "2. Add Item List"
    Display "3. Modify Item List"
    Display "4. Print Invoice"
    Display "5. Quit"
    Display “Choice: ”
    Get select

    Clear the screen by using system(“cls”)
    Ignore one or more characters from get buffer by using cin.ignore()

    If select equal to 1
        Call displayItemList
    Else if select equal to 2
```

```
        Call addItemList
    Else if select equal to 3
        Call modifyItemList
    Else if select equal to 4
        Call printINvoice
    Else if select equal to 5
        Exit the loop by using Break
    Else
        Display “Not available”
    End If
    Call writeItemList
    Display “Continue?  (1-yes, 2-no): “
    Get cont
    Clear the screen by using system(“cls”)
While (cont equal to 1)
End Do
Display total by Call calculateGrandTotal which will return total with 2 digits after decimal point
Return 0
End Function
```

readItemList

```
Function readItemList (purchase_day, purchase_date, goods_name, goods_code, goods_price,
goods_weight, number_goods)
Begin
Declare index = 0

Open “itemList.txt” For read

Get purchase_day from “itemList.txt”
Get purchase_date from “itemList.txt”

While not End of File (EOF)
    Get goods_name[index] from “itemList.txt”
    Get goods_code[index] from “itemList.txt”
    Get goods_price [index] and goods_weight from “itemList.txt”
    Ignore one or more characters from get buffer by using readfile. ignore ()
    Index increment by 1
End While
number_goods = index
Close “itemList.txt”
End Function
```

writeItemList

```
Function writeItemList (purchase_day, purchase_date, goods_name, goods_code, goods_price,
goods_weight, number_goods)
Begin
Open “itemList.txt” For write

Display purchase_day into “itemList.txt”
Display purchase_date into “itemList.txt”

For i in 1 to number_goods but exclude number_goods
    Display goods_name [ i ] into “itemList.txt”
    Display goods_code [i] into “itemList.txt”
    Display goods_price [i] and goods_weight [i] into “itemList.txt” with 2 digits after
    decimal point
    If i is not equal to number_goods minus one
        Display a newline into“itemList.txt”
    End If
End For
Close “itemList.txt”
End Function
```

addItemList

```
Function addItemList (goods_name, goods_code, goods_price, goods_weight, number_goods)

Begin
Declare select [20]
Declare number = 0
Declare alphabet = 0
Declare unknown = 0
Declare not_correct = true

Do
    Display "Add new item? (1-yes, 2-no): "
    Get select with max size 20 character

    For i in 0 to length of select but exclude length of select
        If select[i] is digit
            number increment by 1
        Else if select[i] is alphabet
            alphabet increment by 1
        Else
            unknown increment by 1
        End If
    End For

    For i in 0 to length of select but exclude length of select
```

```
        If number is equal to 1 and alphabet is equal to 0 and unknown is equal to 0
            Switch select [0]
            Case ‘1’:
                not_correct = false
                Exit the switch by using Break
            Case ‘2’:
                not_correct = false
                Exit the switch by using Break
            Default:
                number = 0
                alphabet = 0
                unknown = 0
                not_correct = true
            End switch
        Else
            number = 0
            alphabet = 0
            unknown = 0
            not_correct = true
        End If
    End For
While (not_correct is true)
End Do

If select [0] is equal to ‘1’
    For i in 0 to number_goods but include number_goods
        If i is equal to number_goods
            Display “New item’s name: ”
            Get goods_ name[number_goods]
            Display “New item’s code: ”
            Get goods_code[number_goods]
            Display “Price of new item per KG (RM): ”
            Get goods_price[number_goods]
            Display “Purchasing weight (KG): ”
            Get goods_weight[number_goods]
        End If
    End For
    number_goods increment by 1
End If
End Function
```

modifyItemList

```
Function modifyItemList (goods_name, goods_code, goods_price, goods_weight, number_goods)

Begin
```

```
Declare found_code = 0
Declare select = 0
Declare index_found = 0
Declare modify_code
Do
    Display “Modify item? ( 1 – yes , 2 – no): ”
    Get select
While select is less or equal to 0 Or select is larger than or equal to 3
End Do

If select is equal to 1
    Ignore one or more characters from get buffer by using cin.ignore()
    Display “Key in item code: ”
    Get modify_code

    For i in 0 to number_goods but exclude number_goods
        If goods_code[i] is equal to modify_code
            found_code = 1
            index_found = 1
        End If
    End For

    If found_code is equal to 1
        Display “Item found!”
        Display "New item's name: "
        Get goods_name[index_found]
        Display "New item's code: "
        Get goods_code[index_found]
        Display "Price per KG (RM): "
        Get goods_price[index_found]
        Display "Purchasing Weight (KG): "
        Get goods_weight[index_found]
    Else
        Display "Item is not available."
    End If
    found_code = 0;
End If
End Function
```

displayItemList

```
Function displayItemList (goods_name, goods_code, goods_price, goods_weight, number_goods)
Begin
For i in 0 to number_goods but exclude number_goods
```

```
    Display “Item” i+1
    Display goods_name [i]
    Display goods_code [i]
    Display goods_price [i] with 2 digits after decimal point
    Display goods_weight [i] with 2 digits after decimal point
End For
End Function
```

printInvoice

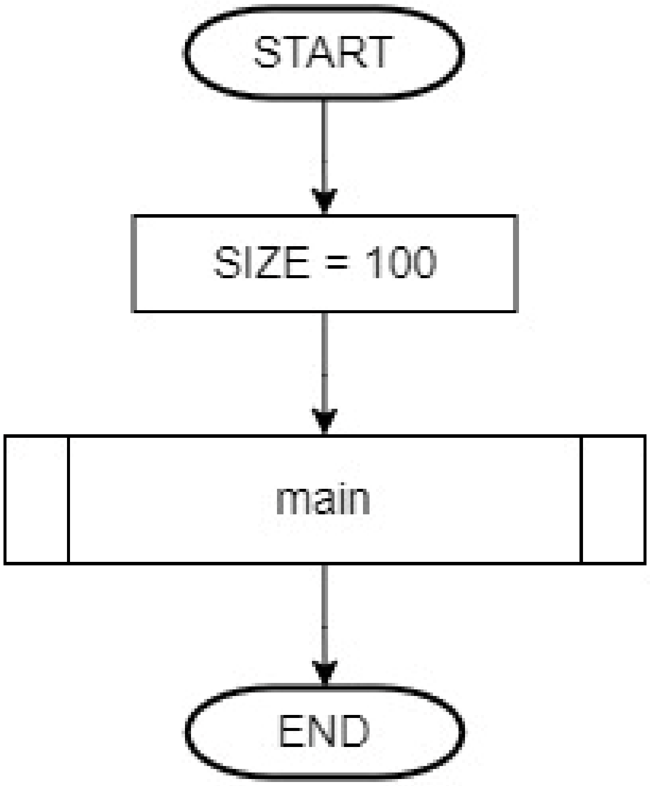
```
Function printInvoice (purchase_day, purchase_date, goods_name, goods_code, goods_price,
goods_weight, number_goods)
Begin
Declare total = 0

Open “invoice.txt” For write

Display purchase_date into “invoice.txt”
Display purchase_day into “invoice.txt”

For i in 1 to number_goods but exclude number_goods
    Display goods_name [ i ] into “invoice.txt”
    Display goods_price [i] , goods_weight [i] and  price for each goods per weight =
    (good_weight[i] * goods_price[i]) into “invoice.txt” with 2 digits after decimal point
    total = total + (good_weight[i] * goods_price[i])
End For
Display total into “invoice.txt”
Close “invoice.txt”
End Function
```

calculateGrandTotal

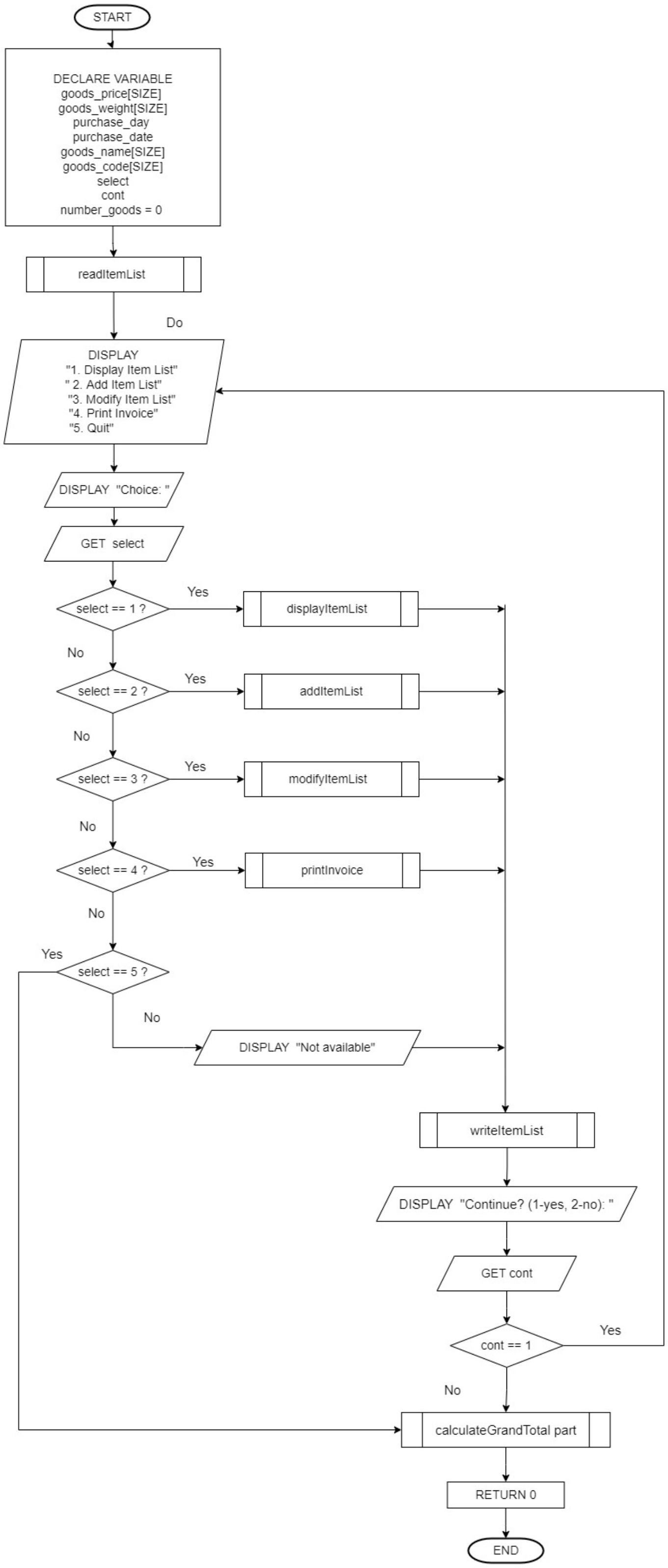
```
Function calculateGrandTotal (goods_price, goods_weight, number_goods)
Begin
Declare total = 0

For i in 1 to number_goods but exclude number_goods
    total = total + (good_weight[i] * goods_price[i])
End For
return total
End Function
```

C. Flowchart

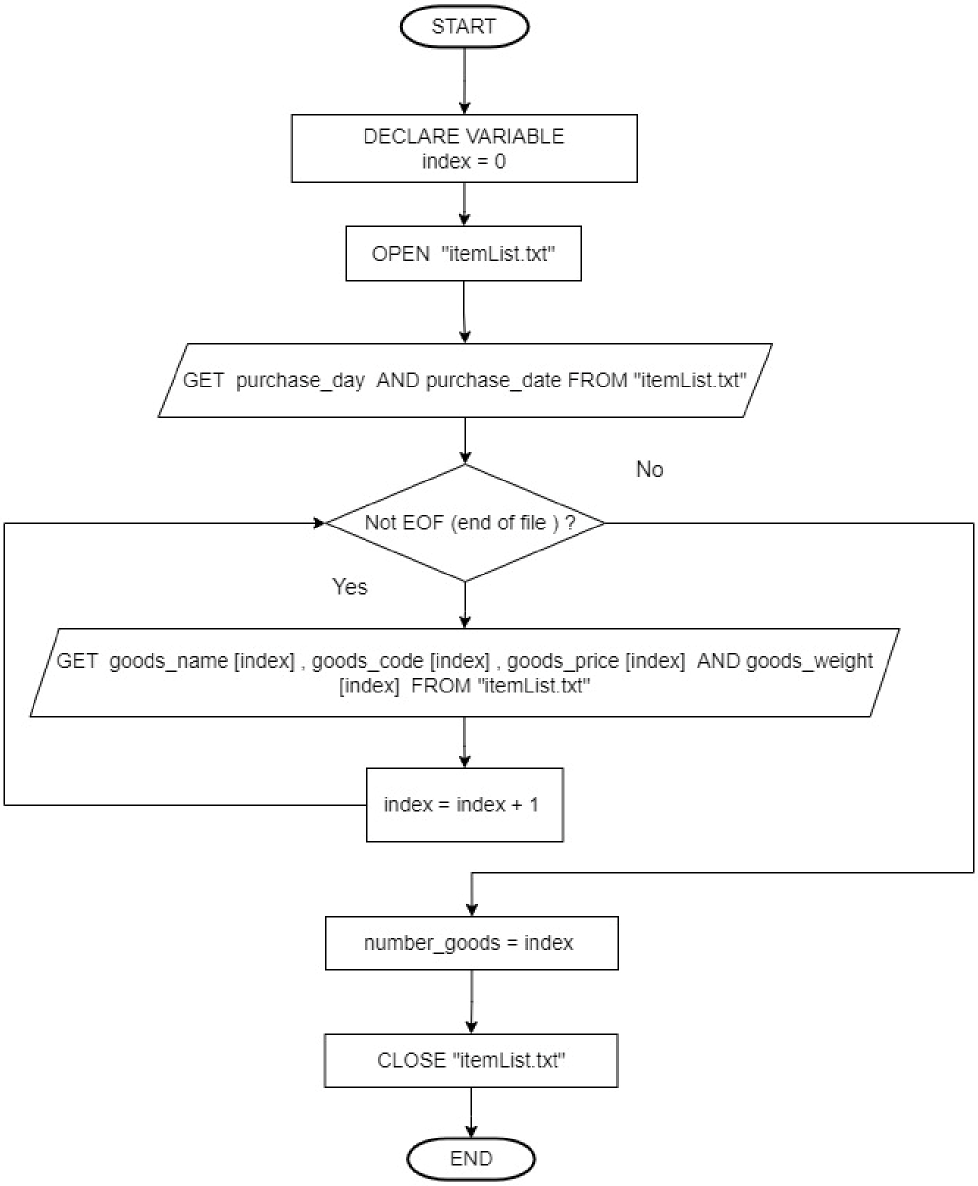
Define Size = 100



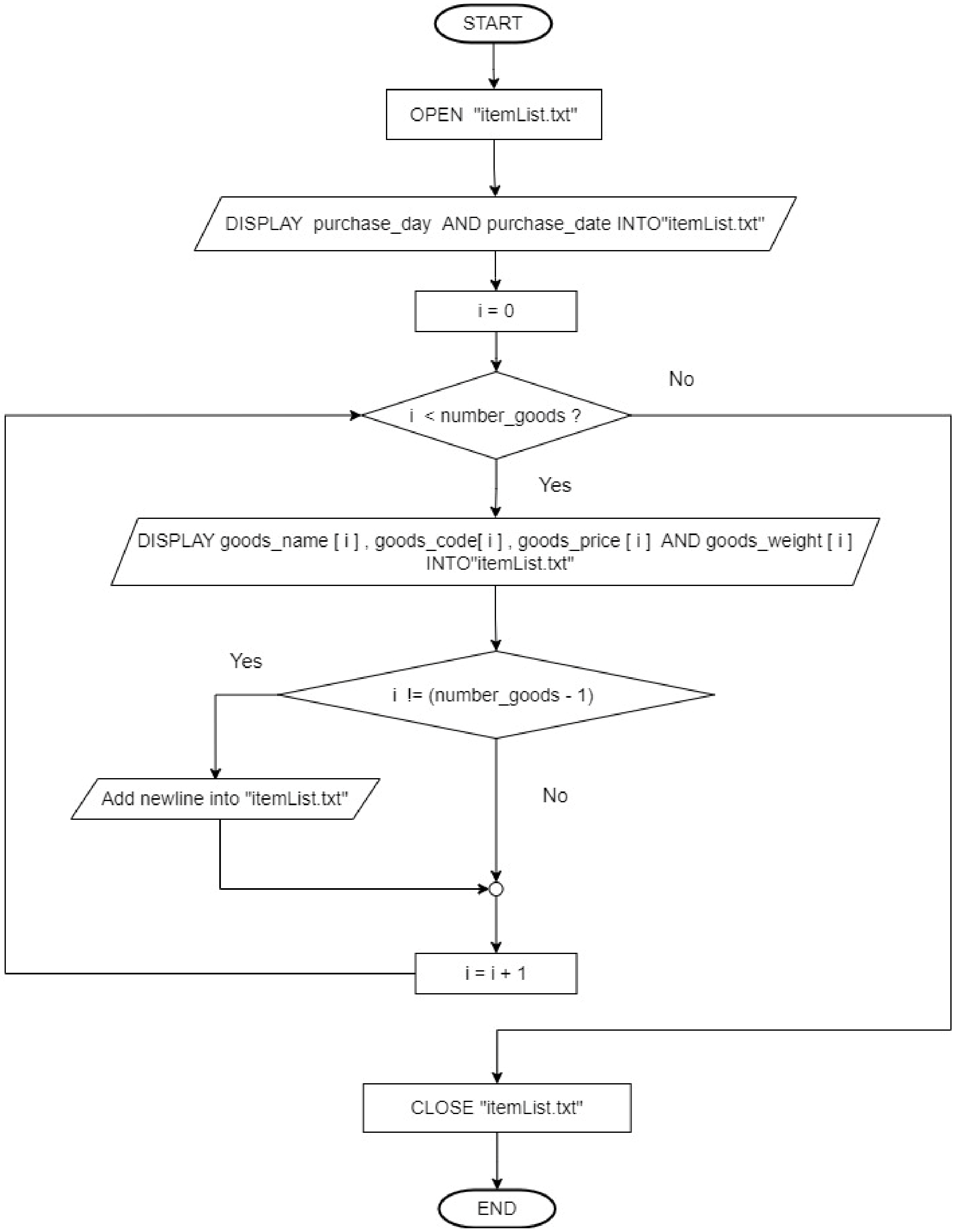
Function Main



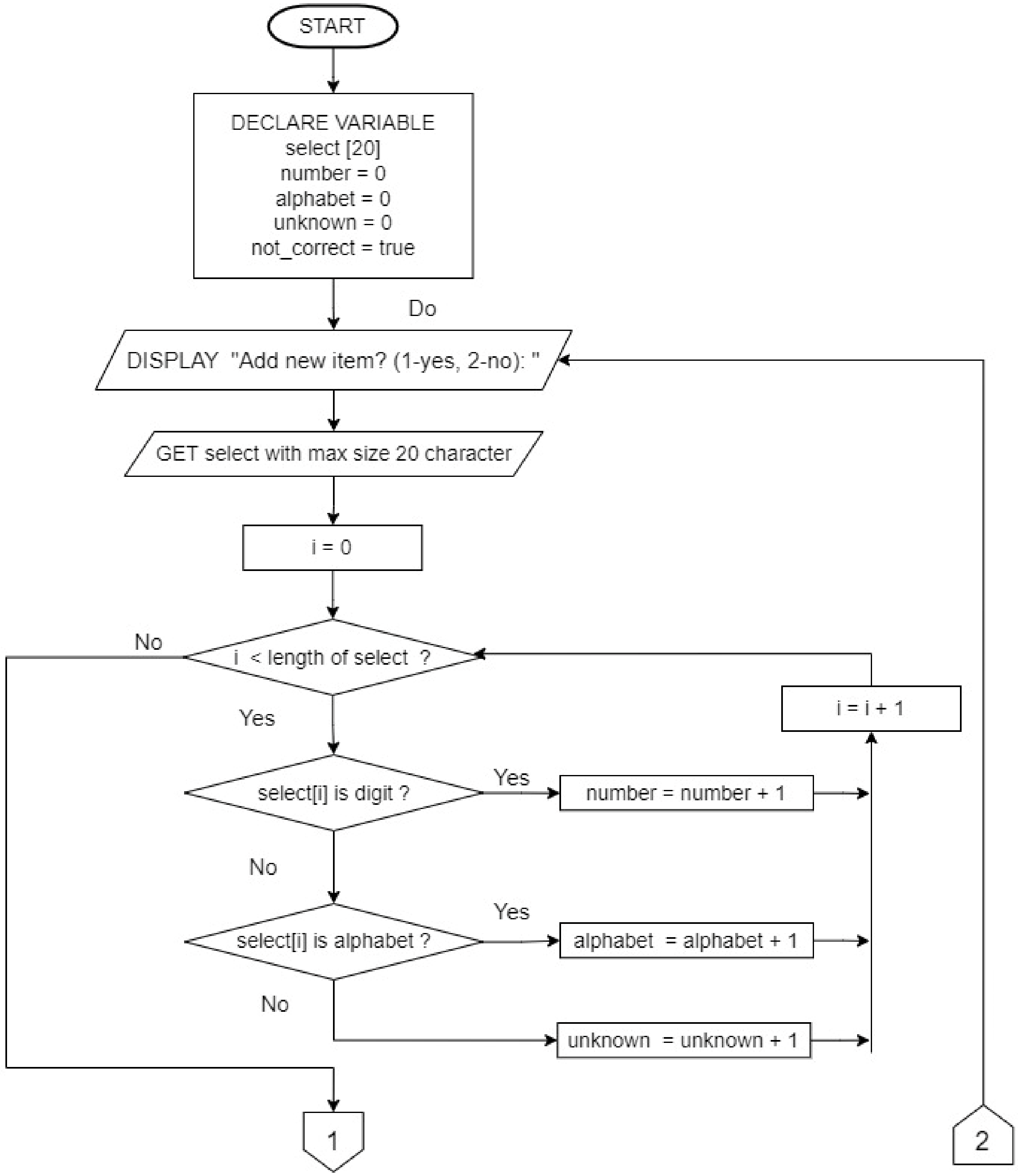
readItemList

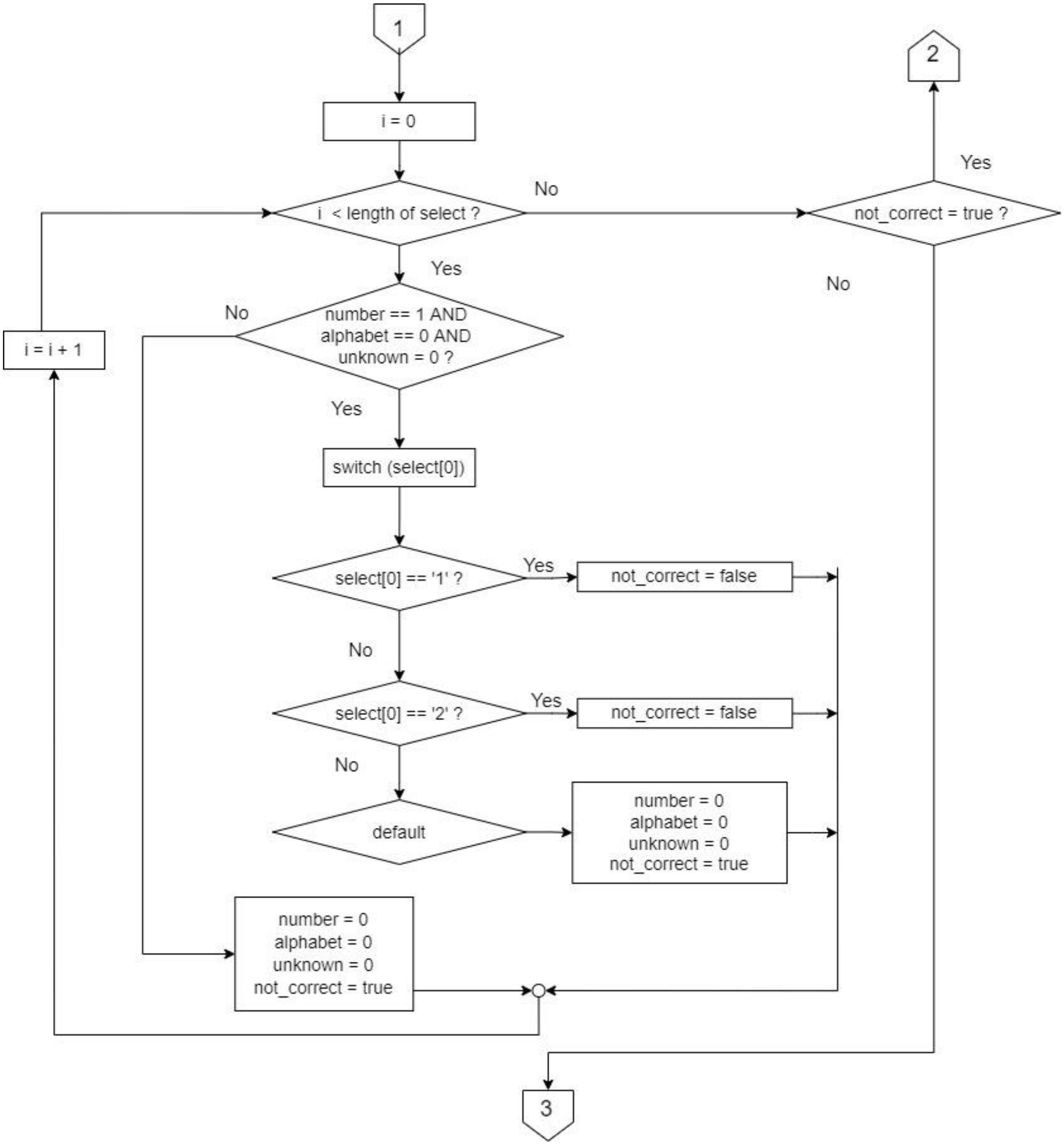

START
DECLARE VARIABLE
index = 0
OPEN "itemList.txt"
GET purchase_day AND purchase_date FROM "itemList.txt"
Not EOF (end of file) ?
No
Yes
GET goods_name [index] , goods_code [index] , goods_price [index] AND goods_weight [index] FROM "itemList.txt"
index = index + 1
number_goods = index
CLOSE "itemList.txt"
END

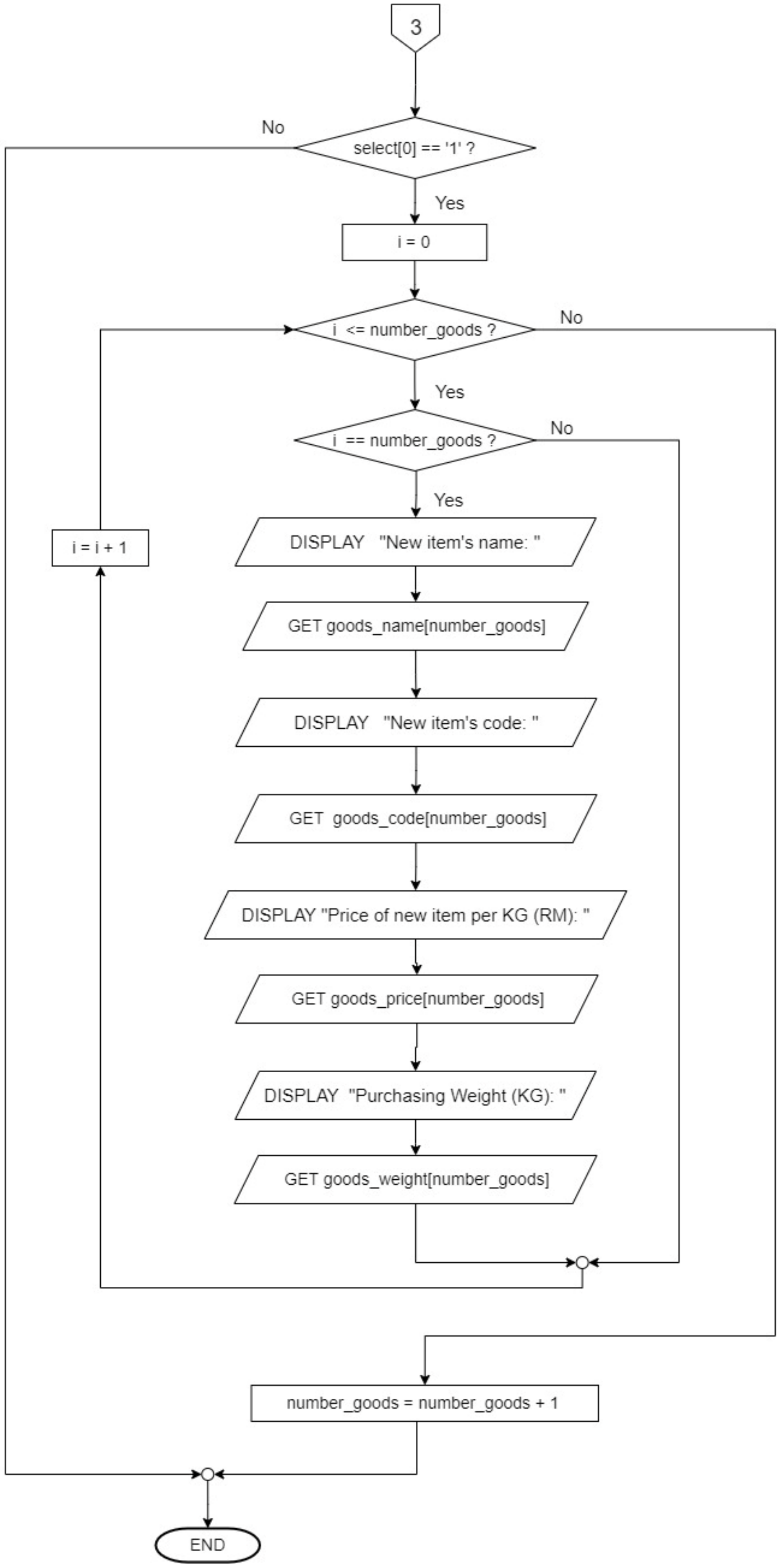
writefileList


START
OPEN "itemList.txt"
DISPLAY purchase_day AND purchase_date INTO"itemList.txt"
i = 0
i < number_goods ?
No
Yes
DISPLAY goods_name [i] , goods_code[i] , goods_price [i] AND goods_weight [i] INTO"itemList.txt"
Yes
i != (number_goods - 1)
Add newline into "itemList.txt"
No
i = i + 1
CLOSE "itemList.txt"
END

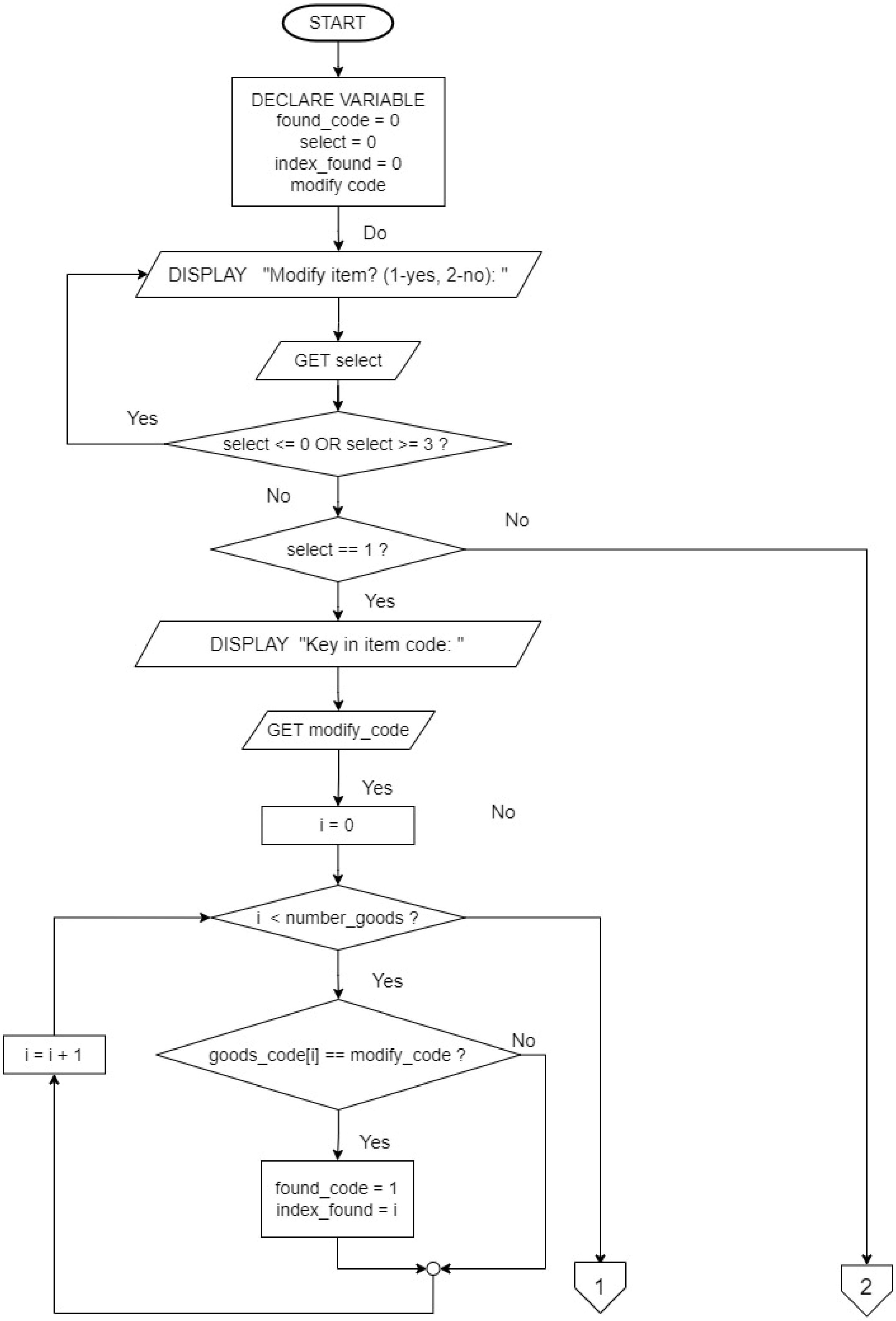
addItemList

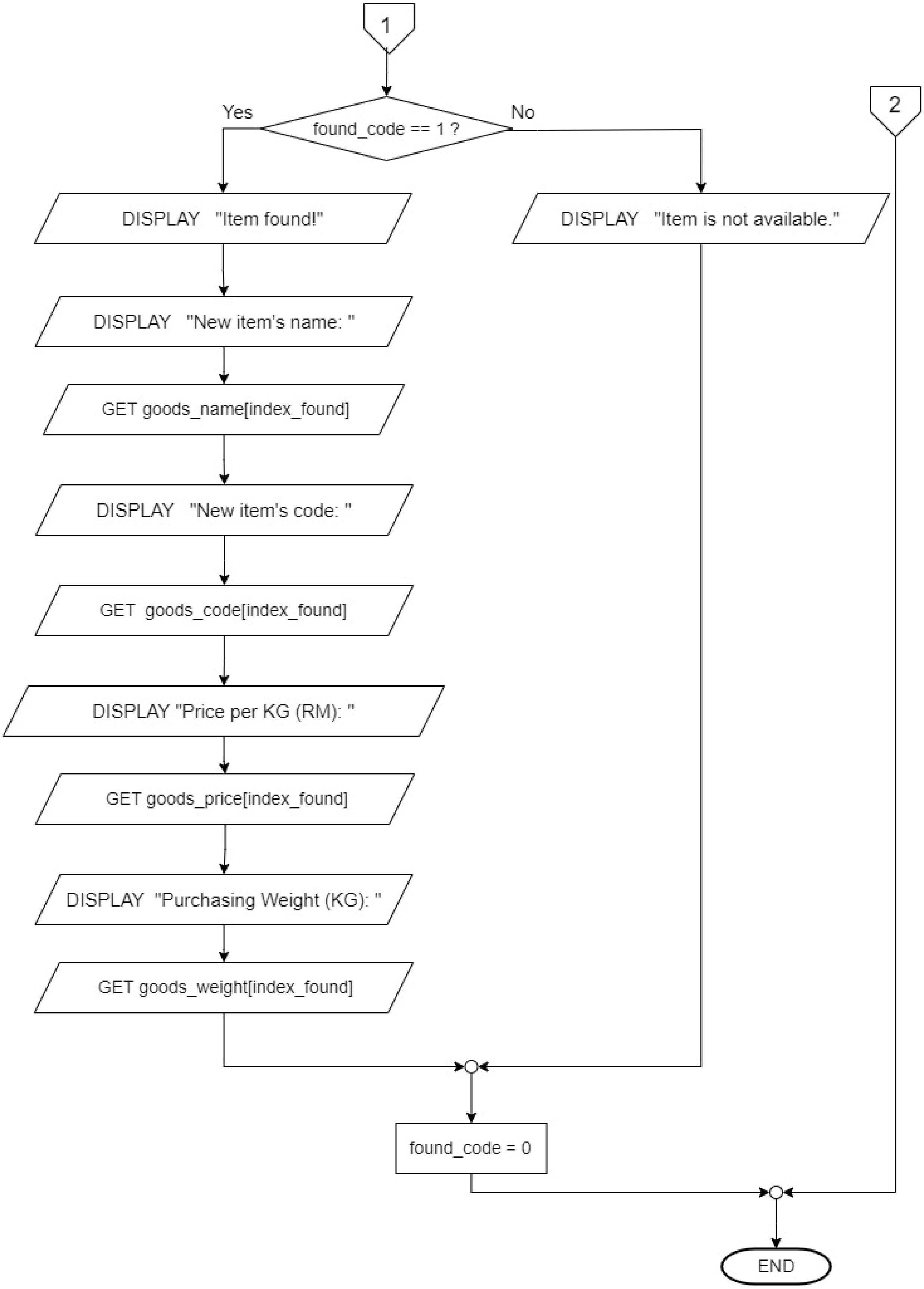



1
i = 0
i < length of select ?
No
Yes
not_correct = true ?
Yes
2
No
No
number == 1 AND
alphabet == 0 AND
unknown = 0 ?
i = i + 1
Yes
switch (select[0])
select[0] == '1' ?
Yes
not_correct = false
No
select[0] == '2' ?
Yes
not_correct = false
No
default
number = 0
alphabet = 0
unknown = 0
not_correct = true
number = 0
alphabet = 0
unknown = 0
not_correct = true
3

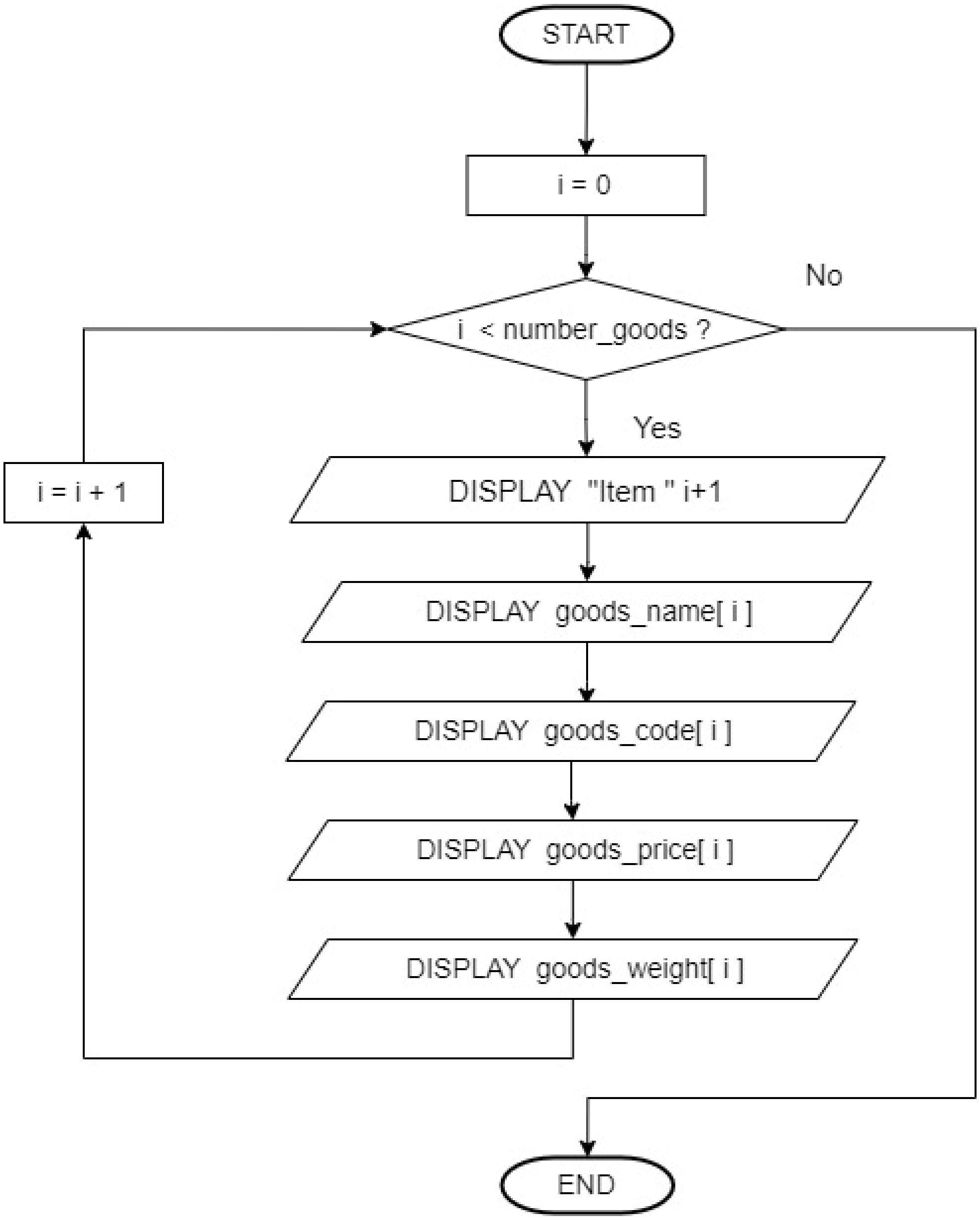

3
select[0] == '1' ?
No
Yes
i = 0
i <= number_goods ?
No
Yes
i == number_goods ?
No
Yes
i = i + 1
DISPLAY "New item's name: "
GET goods_name[number_goods]
DISPLAY "New item's code: "
GET goods_code[number_goods]
DISPLAY "Price of new item per KG (RM): "
GET goods_price[number_goods]
DISPLAY "Purchasing Weight (KG): "
GET goods_weight[number_goods]
number_goods = number_goods + 1
END

modifyItemList

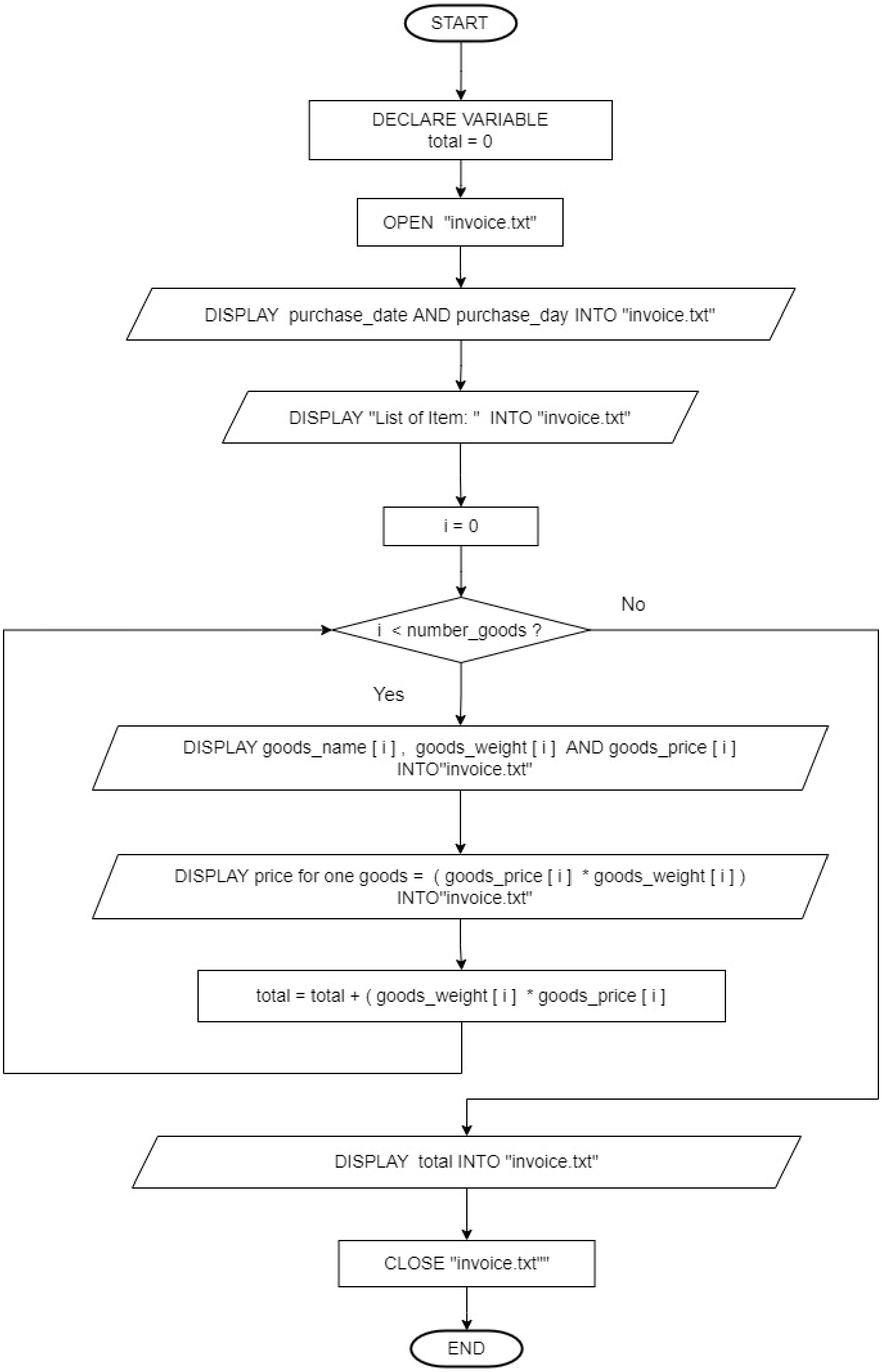



1
found_code == 1 ?
Yes
No
2
DISPLAY "Item found!"
DISPLAY "Item is not available."
DISPLAY "New item's name: "
GET goods_name[index_found]
DISPLAY "New item's code: "
GET goods_code[index_found]
DISPLAY "Price per KG (RM): "
GET goods_price[index_found]
DISPLAY "Purchasing Weight (KG): "
GET goods_weight[index_found]
found_code = 0
END

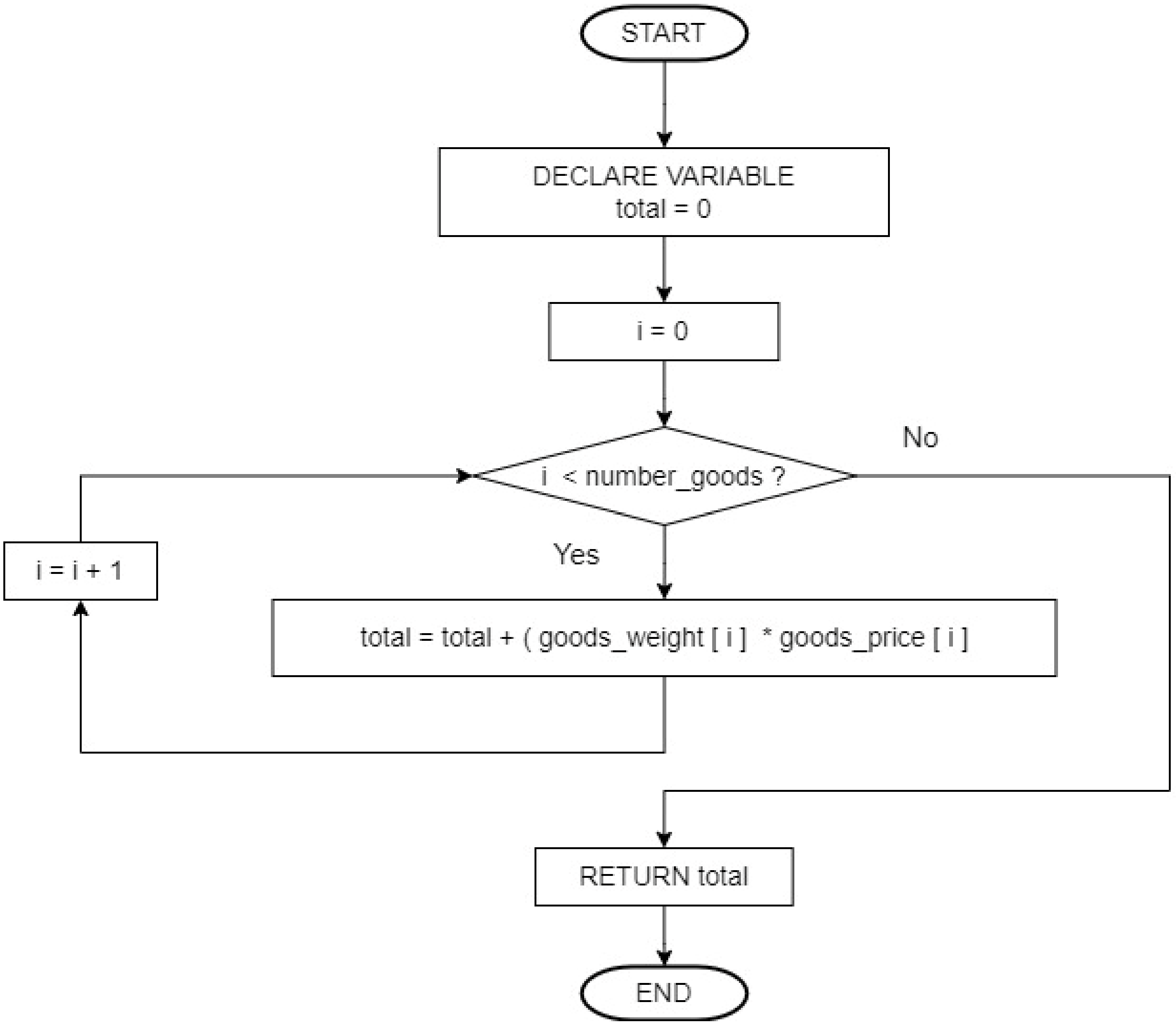
displayItemList



printInvoice


START
DECLARE VARIABLE
total = 0
OPEN "invoice.txt"
DISPLAY purchase_date AND purchase_day INTO "invoice.txt"
DISPLAY "List of Item: " INTO "invoice.txt"
i = 0
i < number_goods ?
No
Yes
DISPLAY goods_name [i] , goods_weight [i] AND goods_price [i]
INTO"invoice.txt"
DISPLAY price for one goods = (goods_price [i] * goods_weight [i])
INTO"invoice.txt"
total = total + (goods_weight [i] * goods_price [i]
DISPLAY total INTO "invoice.txt"
CLOSE "invoice.txt""
END

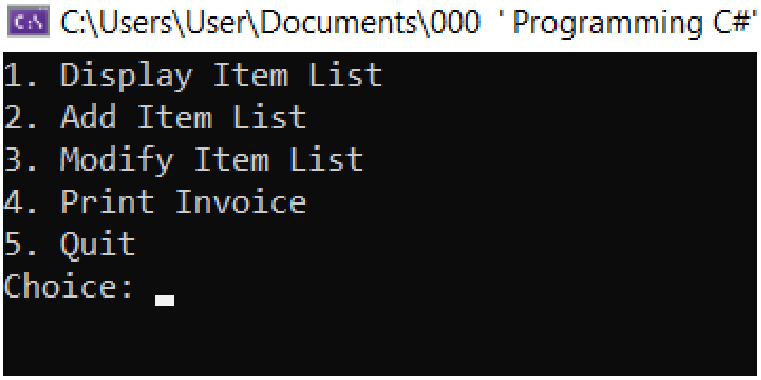
calculateGrandTotal



D. Test case

Test case 1 – Main Menu and Display Item List :

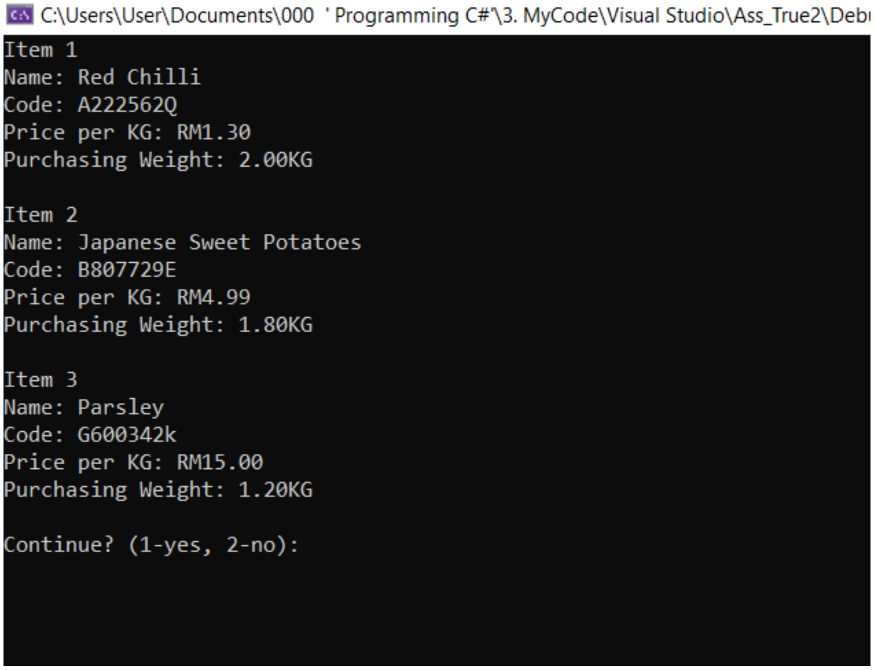
If user executed the program, the program will display the main menu as shown in Figure 1.0 for user to insert the choice.



```
1. Display Item List
2. Add Item List
3. Modify Item List
4. Print Invoice
5. Quit
Choice:
```

Figure 1.0

If user insert the choice equal to 1 , the program will display all the Item List as shown in Figure2.0 .

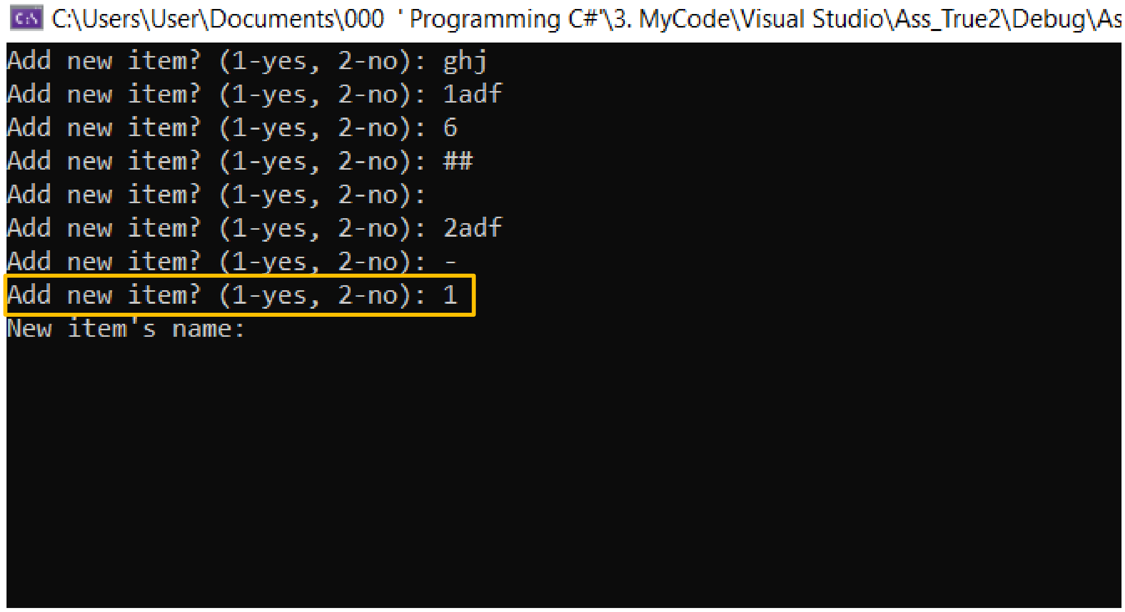

C:\Users\User\Documents\000 ' Programming C#'\3. MyCode\Visual Studio\Ass_True2\Debu

```
Item 1
Name: Red Chilli
Code: A222562Q
Price per KG: RM1.30
Purchasing Weight: 2.00KG

Item 2
Name: Japanese Sweet Potatoes
Code: B807729E
Price per KG: RM4.99
Purchasing Weight: 1.80KG

Item 3
Name: Parsley
Code: G600342k
Price per KG: RM15.00
Purchasing Weight: 1.20KG

Continue? (1-yes, 2-no):
```

Figure 2.0

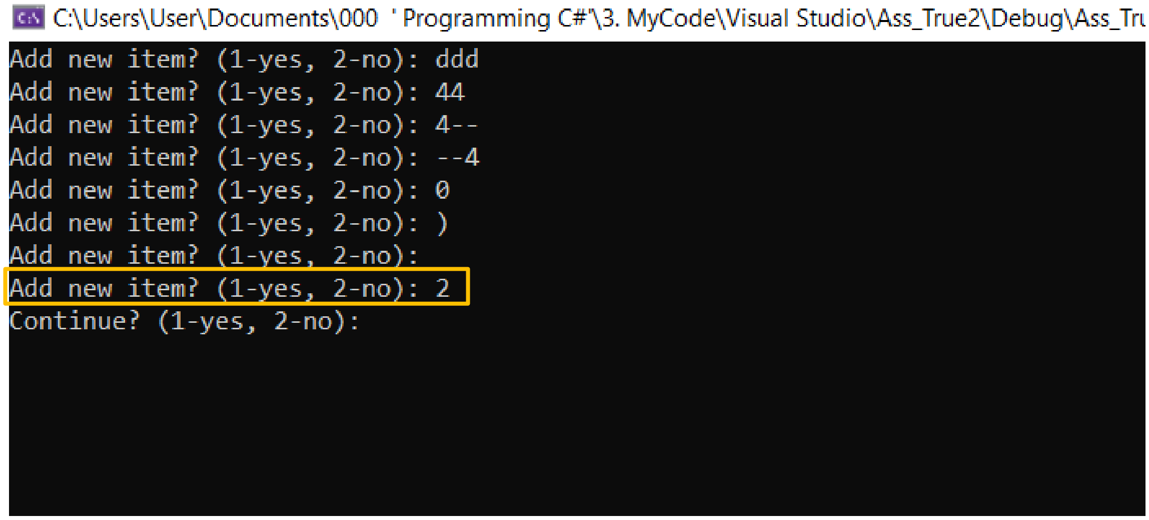
Test case 2 - Add Item List

If user insert the choice is 2 which is Add Item List in main menu, the program will display “Add new item? (1-yes, 2 no): ” for user to insert. If the value that user insert is not 1 or 2, the program will display again until user insert the correct one as shows in Figure 3.0 and Figure 3.1.


C:\Users\User\Documents\000 ' Programming C#'\3. MyCode\Visual Studio\Ass_True2\Debug\As

```
Add new item? (1-yes, 2-no): ghj
Add new item? (1-yes, 2-no): 1adf
Add new item? (1-yes, 2-no): 6
Add new item? (1-yes, 2-no): ##
Add new item? (1-yes, 2-no):
Add new item? (1-yes, 2-no): 2adf
Add new item? (1-yes, 2-no): -
Add new item? (1-yes, 2-no): 1
New item's name:
```

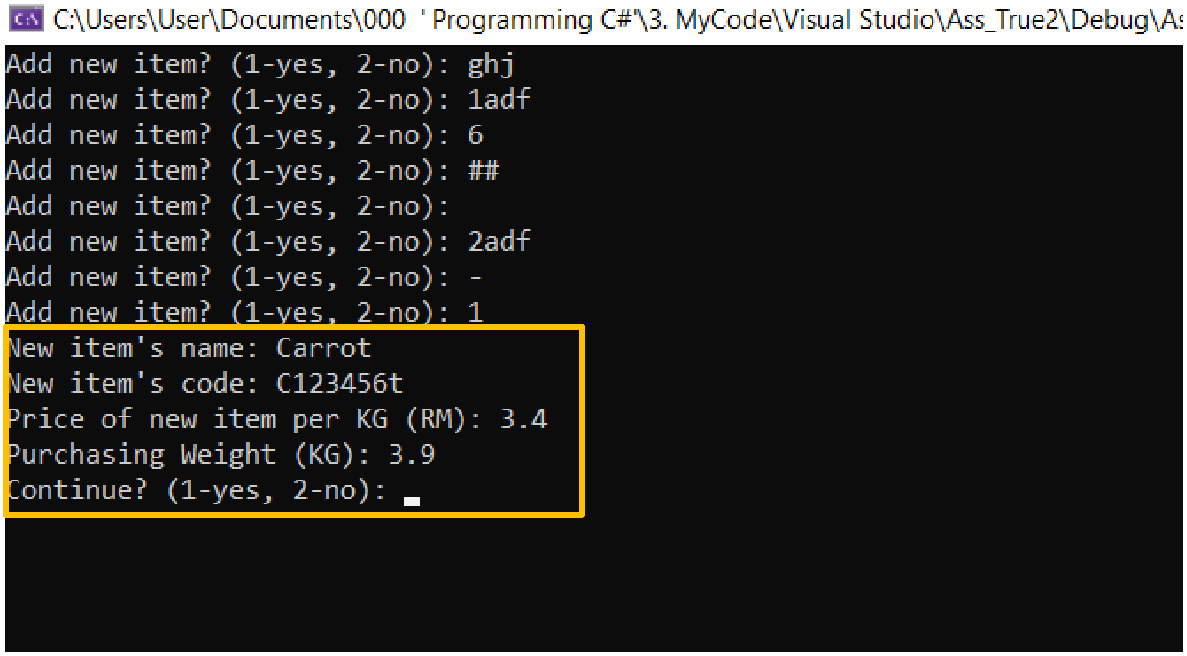
Figure 3.0


C:\Users\User\Documents\000 ' Programming C#'\3. MyCode\Visual Studio\Ass_True2\Debug\Ass_Tru

```
Add new item? (1-yes, 2-no): ddd
Add new item? (1-yes, 2-no): 44
Add new item? (1-yes, 2-no): 4--
Add new item? (1-yes, 2-no): --4
Add new item? (1-yes, 2-no): 0
Add new item? (1-yes, 2-no): )
Add new item? (1-yes, 2-no):
Add new item? (1-yes, 2-no): 2
Continue? (1-yes, 2-no):
```

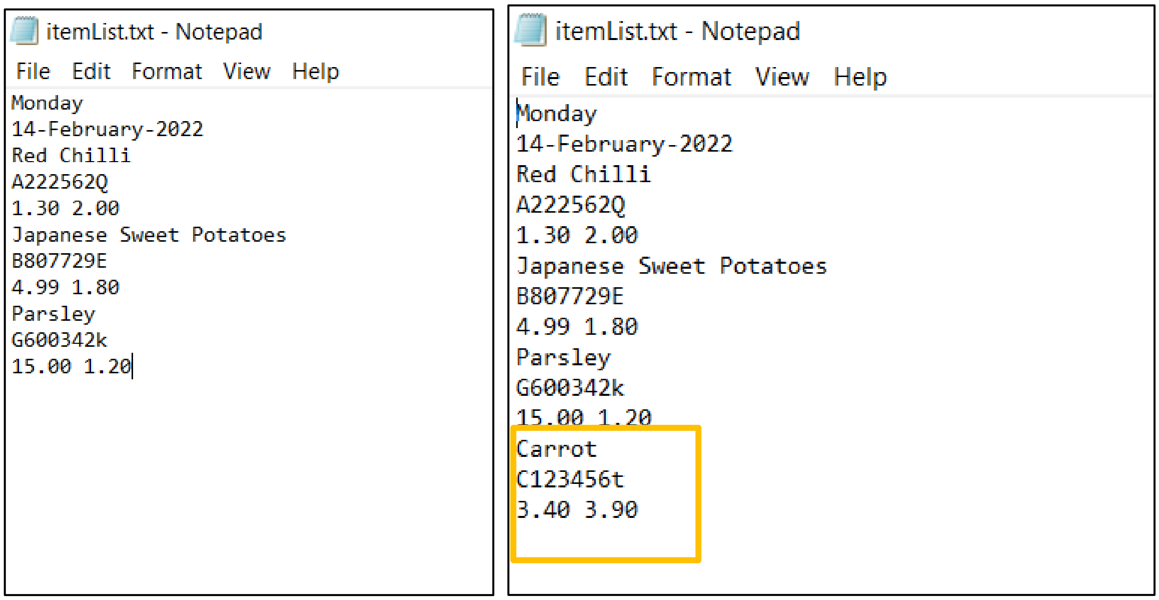
Figure 3.1

If user insert correctly and it is 1, the program will let user to insert the new item’s name, new item’s code, price of new item (KG) and item weight (RM) as shown in Figure 3.2. Once the user inserts all the information of the new item, the text file named “itemList.txt” will automatically update it as shown in Figure 3.3. Besides, the Display Item List in main menu also will update as shown in Figure 3.4.


C:\Users\User\Documents\000 ' Programming C#'\3. MyCode\Visual Studio\Ass_True2\Debug\A

```
Add new item? (1-yes, 2-no): ghj
Add new item? (1-yes, 2-no): 1adf
Add new item? (1-yes, 2-no): 6
Add new item? (1-yes, 2-no): ##
Add new item? (1-yes, 2-no):
Add new item? (1-yes, 2-no): 2adf
Add new item? (1-yes, 2-no): -
Add new item? (1-yes, 2-no): 1
New item's name: Carrot
New item's code: C123456t
Price of new item per KG (RM): 3.4
Purchasing Weight (KG): 3.9
Continue? (1-yes, 2-no): _
```

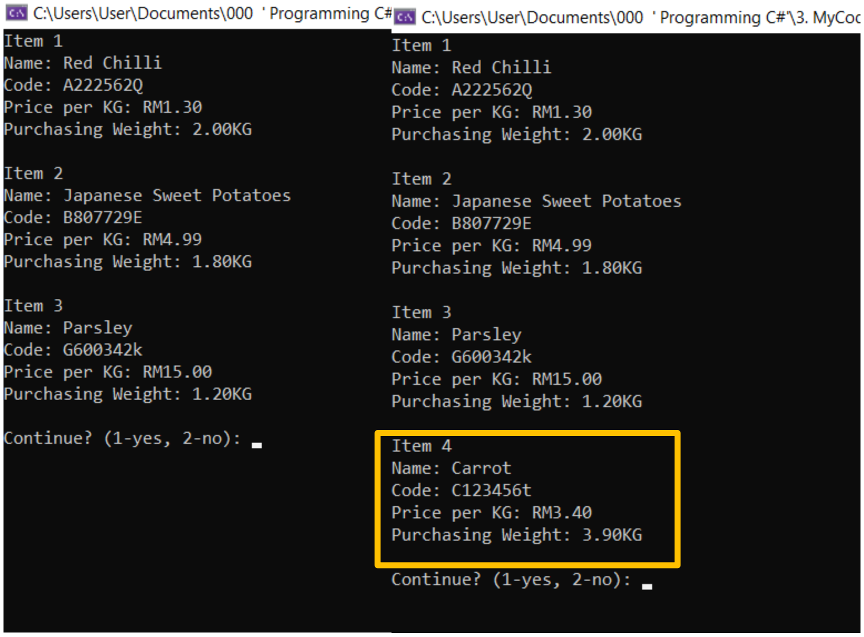
Figure 3.2 : Add Information of Item



```
Monday
14-February-2022
Red Chilli
A222562Q
1.30 2.00
Japanese Sweet Potatoes
B807729E
4.99 1.80
Parsley
G600342k
15.00 1.20
```

```
Monday
14-February-2022
Red Chilli
A222562Q
1.30 2.00
Japanese Sweet Potatoes
B807729E
4.99 1.80
Parsley
G600342k
15.00 1.20
Carrot
C123456t
3.40 3.90
```

Figure 3.3 : Left is before adding item and Right is after adding item in “itemList.txt”

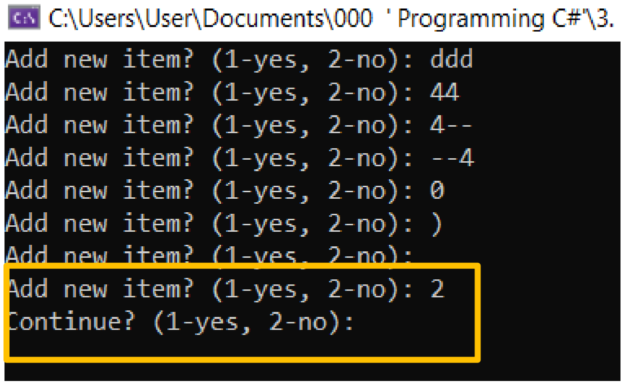

C:\Users\User\Documents\000 ' Programming C#
C:\Users\User\Documents\000 ' Programming C#'\3. MyCoc

```
Item 1
Name: Red Chilli
Code: A222562Q
Price per KG: RM1.30
Purchasing Weight: 2.00KG

Item 2
Name: Japanese Sweet Potatoes
Code: B807729E
Price per KG: RM4.99
Purchasing Weight: 1.80KG

Item 3
Name: Parsley
Code: G600342k
Price per KG: RM15.00
Purchasing Weight: 1.20KG

Continue? (1-yes, 2-no):
```

```
Item 1
Name: Red Chilli
Code: A222562Q
Price per KG: RM1.30
Purchasing Weight: 2.00KG

Item 2
Name: Japanese Sweet Potatoes
Code: B807729E
Price per KG: RM4.99
Purchasing Weight: 1.80KG

Item 3
Name: Parsley
Code: G600342k
Price per KG: RM15.00
Purchasing Weight: 1.20KG

Item 4
Name: Carrot
Code: C123456t
Price per KG: RM3.40
Purchasing Weight: 3.90KG

Continue? (1-yes, 2-no):
```

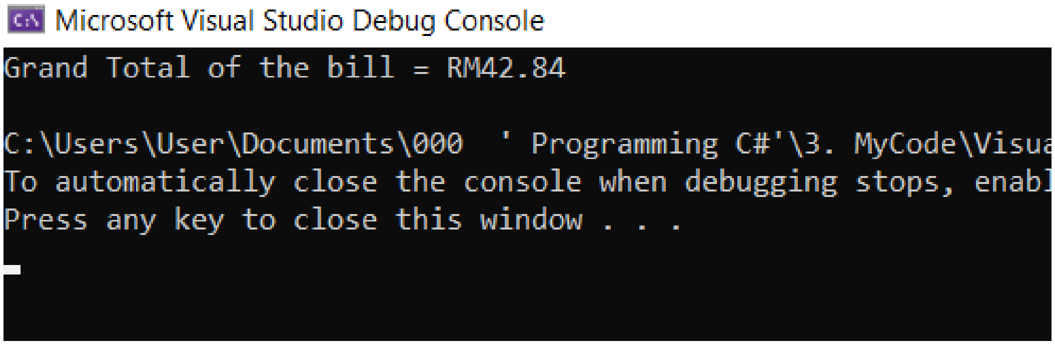
Figure 3.4 Left is before adding item and Right is after adding item in Display Item List

If user insert correctly and it is 2, the program will display “Continue? (1-yes, 2-no) as shown in Figure 3.5 and let user choose whether continue the program or stop it. If choose 1 which continue it will back to the main menu. If choose 2 the system will directly show the grand total of all items as shown 3.6.



```
Add new item? (1-yes, 2-no): ddd
Add new item? (1-yes, 2-no): 44
Add new item? (1-yes, 2-no): 4--
Add new item? (1-yes, 2-no): --4
Add new item? (1-yes, 2-no): 0
Add new item? (1-yes, 2-no): )
Add new item? (1-yes, 2-no):
Add new item? (1-yes, 2-no): 2
Continue? (1-yes, 2-no):
```

Figure 3.5



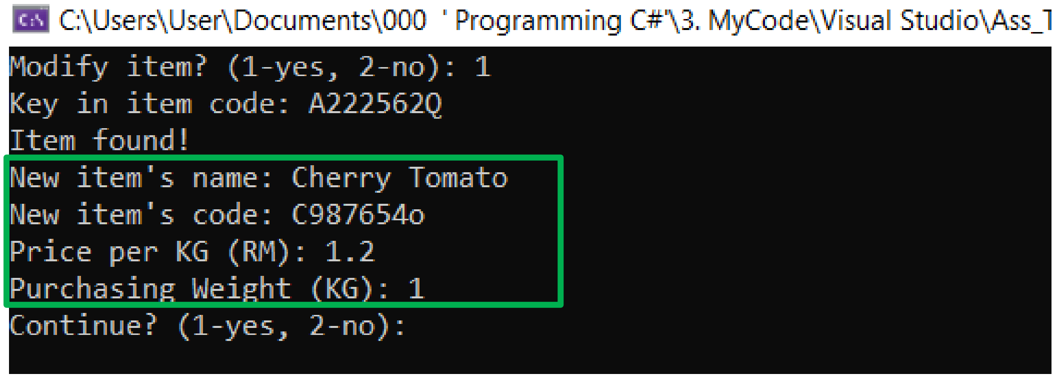
```
Grand Total of the bill = RM42.84

C:\Users\User\Documents\000  ' Programming C#'\3. MyCode\Visua
To automatically close the console when debugging stops, enabl
Press any key to close this window . . .
```

Figure 3.6: This grand total includes the added item (Carrot)

Test case 3 – Modify Item List

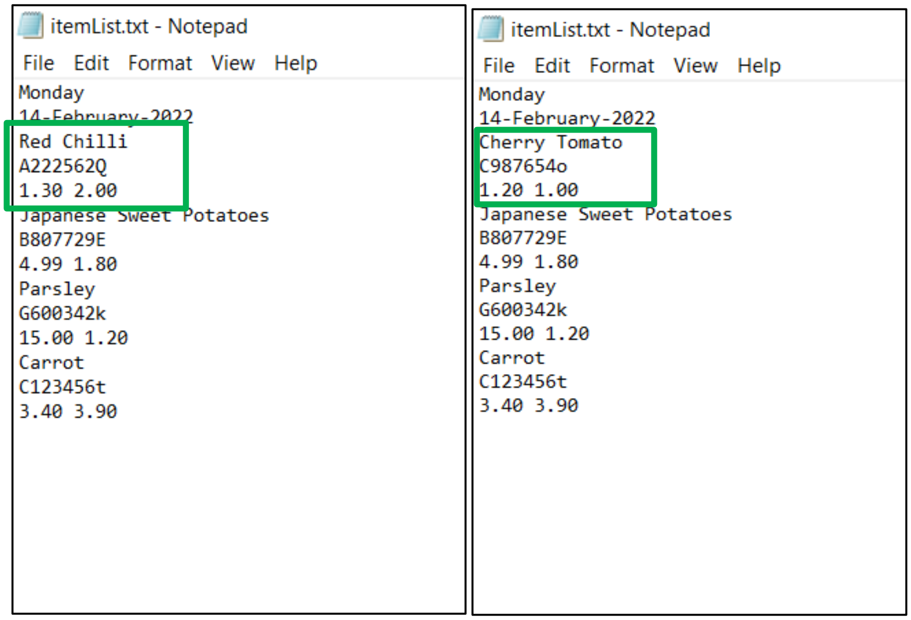
If user insert the choice is 3 which is Modify Item List in main menu, the program will display “Modify item? (1-yes, 2 no):” for user to insert.

If user choice 1-yes, the program will need user to insert the code of item. Once the code of item matches with the code of item in item list, the program will let user to insert the information of item such as item’s name, item’s code, price per KG (RM) and purchasing Weight (KG) as shown in Figure 4.0. The program will also modify the information of item in text file named “itemList.txt” as shown in Figure 4.1. Besides, the Display Item List in main menu also will update as shown in Figure 4.2. Once the code of item does not match, the program will display “Item is not available” as shown in Figure 4.3.


C:\Users\User\Documents\000 ' Programming C#'\3. MyCode\Visual Studio\Ass_1

```
Modify item? (1-yes, 2-no): 1
Key in item code: A222562Q
Item found!
New item's name: Cherry Tomato
New item's code: C987654o
Price per KG (RM): 1.2
Purchasing Weight (KG): 1
Continue? (1-yes, 2-no):
```

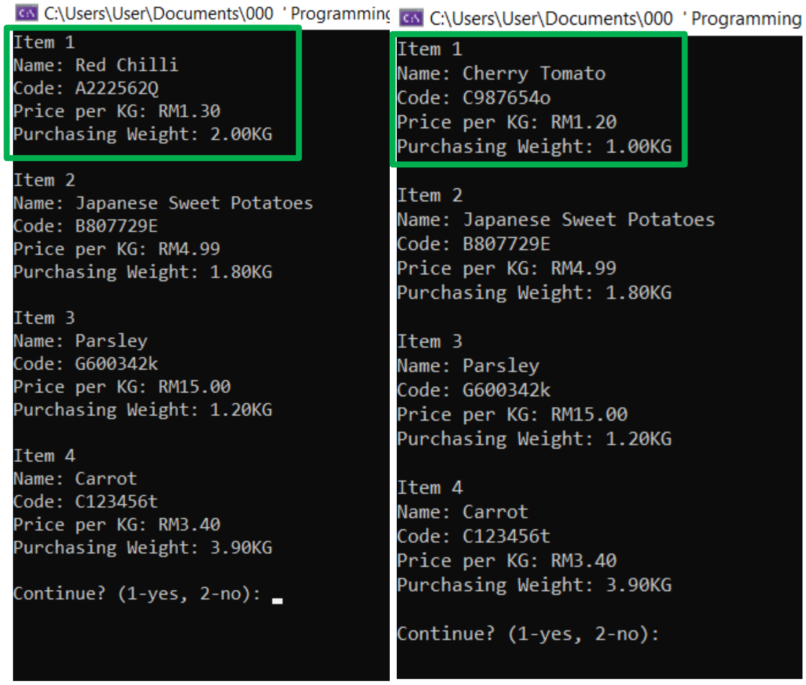
Figure 4.0: If code of item is matches with the codes in item list



```
Monday
14-February-2022
Red Chilli
A222562Q
1.30 2.00
Japanese Sweet Potatoes
B807729E
4.99 1.80
Parsley
G600342k
15.00 1.20
Carrot
C123456t
3.40 3.90
```

```
Monday
14-February-2022
Cherry Tomato
C987654o
1.20 1.00
Japanese Sweet Potatoes
B807729E
4.99 1.80
Parsley
G600342k
15.00 1.20
Carrot
C123456t
3.40 3.90
```

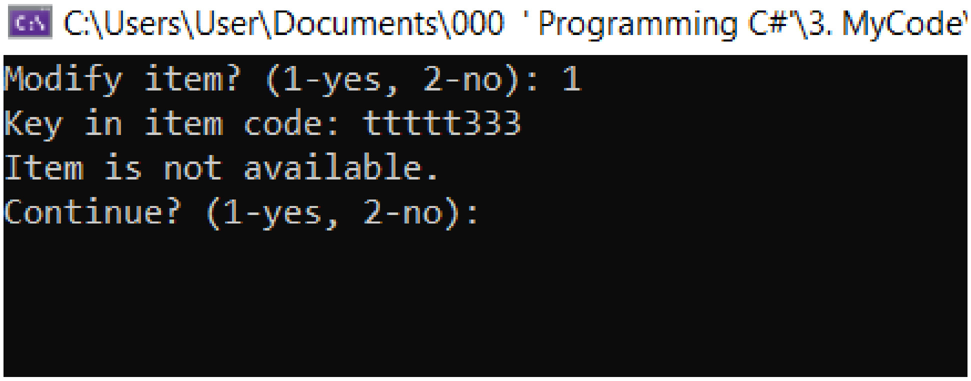
Figure 4.1 : left is before modify and right is after modify in itemList.txt



```
C:\Users\User\Documents\000  ' Programmin
Item 1
Name: Red Chilli
Code: A222562Q
Price per KG: RM1.30
Purchasing Weight: 2.00KG

Item 2
Name: Japanese Sweet Potatoes
Code: B807729E
Price per KG: RM4.99
Purchasing Weight: 1.80KG

Item 3
Name: Parsley
Code: G600342k
Price per KG: RM15.00
Purchasing Weight: 1.20KG

Item 4
Name: Carrot
Code: C123456t
Price per KG: RM3.40
Purchasing Weight: 3.90KG

Continue? (1-yes, 2-no): _
```

```
C:\Users\User\Documents\000  ' Programming
Item 1
Name: Cherry Tomato
Code: C987654o
Price per KG: RM1.20
Purchasing Weight: 1.00KG

Item 2
Name: Japanese Sweet Potatoes
Code: B807729E
Price per KG: RM4.99
Purchasing Weight: 1.80KG

Item 3
Name: Parsley
Code: G600342k
Price per KG: RM15.00
Purchasing Weight: 1.20KG

Item 4
Name: Carrot
Code: C123456t
Price per KG: RM3.40
Purchasing Weight: 3.90KG

Continue? (1-yes, 2-no):
```

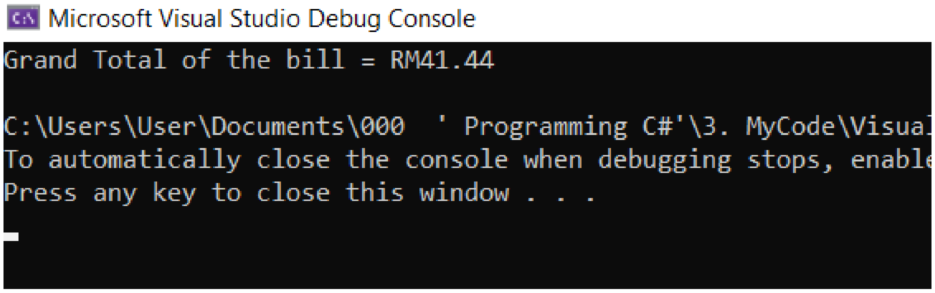
Figure 4.2 : left is before modify and right is after modify in Display Item List



```
C:\Users\User\Documents\000  ' Programming C#'\3. MyCode'
Modify item? (1-yes, 2-no): 1
Key in item code: ttttt333
Item is not available.
Continue? (1-yes, 2-no):
```

Figure 4.3 : If code of item not matches the code in Item List

If user choice 2-no or finish modifying the item, the program will display “Continue? (1-yes, 2 no). If insert yes the program will display the main menu, If choose no , it will display grand total of all item as shown in Figure 4.3.

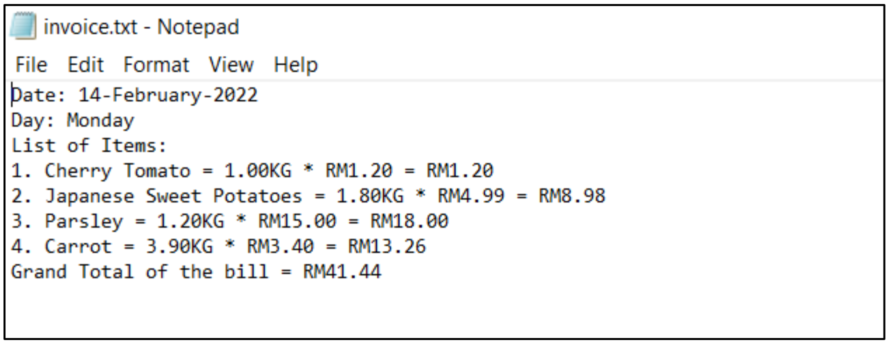


```
Grand Total of the bill = RM41.44

C:\Users\User\Documents\000  ' Programming C#'\3. MyCode\Visua
To automatically close the console when debugging stops, enabl
Press any key to close this window . . .
```

Figure 4.4 : Grand total of all items after added New Item in Test case 2 and after modify in Test case 3

Test case 4 – Print Invoice

If user insert the choice is 3 which is Print Invoice in main menu, the program will create a new file named “invoice.txt” that include all the information of item, purchase date, purchase day, price for one item and grand total of all items as shown in Figure 5.0. After created, the program will display “Continue? (1-yes, 2 no) for user to choose either continue the program or exit the program and display the grand total of all the items.

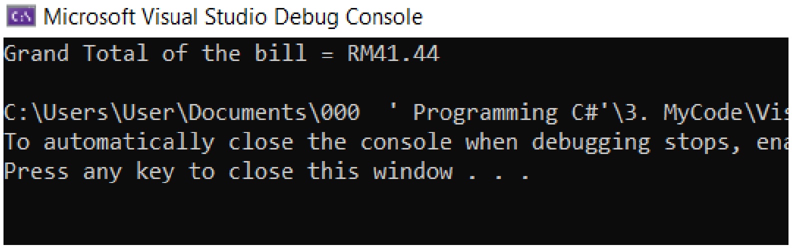


```
Date: 14-February-2022
Day: Monday
List of Items:
1. Cherry Tomato = 1.00KG * RM1.20 = RM1.20
2. Japanese Sweet Potatoes = 1.80KG * RM4.99 = RM8.98
3. Parsley = 1.20KG * RM15.00 = RM18.00
4. Carrot = 3.90KG * RM3.40 = RM13.26
Grand Total of the bill = RM41.44
```

Figure 5.0 : Invoice .txt

Test case 5 – Quit

Once the user insert choice is 5 which is Quit in main menu, the program will stop and display Grand total of the bill as shown in Figure 6.0.



```
Grand Total of the bill = RM41.44

C:\Users\User\Documents\000  ' Programming C#'\3. MyCode\Vi
To automatically close the console when debugging stops, ena
Press any key to close this window . . .
```

Figure 6.0

E. Source Code

```
/* Important notices for the A2 template:
A. You will need to use this template to work on the Assignment 2 (A2)
B. You are not allowed to change the whole structure of template including
adding additional functions or remove any existing functions
C. You will only need to fill in the codes with the parts that indicated
with numbers and its descriptions
D. You will need to ensure all the COUT are exactly the same as shown in A2
doc as the codes are auto marked by a system. LOW marks or ZERO
   mark will be awarded if the system is unable to detect the correct COUT.
E. No extra decorations are allowed as you will be using the template to
work on A2
F. You will need to define the parameters accordingly (including its
datatype) as described below:
   P1 = parameter to store the day
   P2 = parameter to store the date
   P3 = array to store the name of the items
   P4 = array to store the code of the items
   P5 = array to store the price of the items per kilogram
   P6 = array to store the purchasing weight provided by the user
   P7 = parameter to store the number of items available in the list
*/

//Only these libraries are allowed to be defined and used
#include <iostream>
#include <iomanip>
#include <fstream>
#include <string>

using namespace std;

#define SIZE 100 //global definition to determine the size of the array in
A2
//1. Define the FUNCTION PROTOTYPE for all the listed functions
void readItemList(string&, string&, string[], string[], double[], double[],
int&);
void writeItemList(string&, string&, string[], string[], double[],
double[], int&);
void addItemList(string[], string[], double[], double[], int&);
void modifyItemList(string[], string[], double[], double[], int&);
void displayItemList(string[], string[], double[], double[], int&);
void printInvoice(string&, string&, string[], string[], double[], double[],
int&);
double calculateGrandTotal(double[], double[], int&);

int main() {
      //2. Define all the necessary variables
      double goods_price[SIZE], goods_weight[SIZE];
      string purchase_day, purchase_date, goods_name[SIZE],
goods_code[SIZE];
```

```
    int select, cont, number_goods = 0;

    //Call readItemList with the required parameters
    readItemList(purchase_day, purchase_date, goods_name, goods_code,
goods_price, goods_weight, number_goods);

    do {//do...while iteration is implemented to repeat the selection
menu as below
        cout << "1. Display Item List" << endl;
        cout << "2. Add Item List" << endl;
        cout << "3. Modify Item List" << endl;
        cout << "4. Print Invoice" << endl;
        cout << "5. Quit" << endl;

        //User will key in and select one of the functions available
        cout << "Choice: ";
        cin >> select;

        //"cls" stand for clear screen which refreshes the screen,
placing the cursor on original place.
        //For more information, refer to https://www.quora.com/What-is-
system-CLS-for-in-c++
        system("cls");

        //cin.ignore() is to ignore or clear one or more characters
from the input buffer
        //For more information, refer to
https://www.tutorialspoint.com/what-is-the-use-of-cin-ignore-in-cplusplus
        cin.ignore();

        //3. if...else is implemented to select the function according
to the user input
        if (select == 1)
            displayItemList(goods_name, goods_code, goods_price,
goods_weight, number_goods);
        else if (select == 2)
            addItemList(goods_name, goods_code, goods_price,
goods_weight, number_goods);
        else if (select == 3)
            modifyItemList(goods_name, goods_code, goods_price,
goods_weight, number_goods);
        else if (select == 4)
            printInvoice(purchase_day, purchase_date, goods_name,
goods_code, goods_price, goods_weight, number_goods);
        else if (select == 5)
            break;
        else
            cout << "Not available" << endl;
            //Update the items in the text file whatever function is
carried out
            writeItemList(purchase_day, purchase_date, goods_name,
goods_code, goods_price, goods_weight, number_goods);

            //To check whether user wants to continue to perform these
functions or not
            cout << "Continue? (1-yes, 2-no): ";
            cin >> cont;

            system("cls");
      } while (cont == 1); //the do...while will stop if the condition is
false (user choose to stop the program)

      /*4. Grand total of the items is calculated through the RETURN
function */
      cout << "Grand Total of the bill = RM" << fixed << setprecision(2) <<
calculateGrandTotal(goods_price, goods_weight, number_goods) << endl;

      return 0;
}

//5. Function readItemList --> read (ifstream) all the items listed in
"itemList.txt"
//   and store the data into the respective parameters.
//   Hint: P1, P2 and P7 should be reference parameters in this function
void readItemList(string& purchase_day, string& purchase_date,
      string goods_name[], string goods_code[], double goods_price[],
double goods_weight[], int& number_goods) {

      int index = 0;

      ifstream readfile;
      readfile.open("itemList.txt");

      getline(readfile, purchase_day);
      getline(readfile, purchase_date);

      while (!readfile.eof()) {
            getline(readfile, goods_name[index]);
            getline(readfile, goods_code[index]);
            readfile >> goods_price[index] >> goods_weight[index];
            readfile.ignore();
            index++;
      }
      number_goods = index;
      readfile.close();
}
```

```
//6. Function writeItemList --> write / update (ofstream) all the items to
"itemList.txt"
void writeItemList(string& purchase_day, string& purchase_date,
      string goods_name[], string goods_code[], double goods_price[],
double goods_weight[], int& number_goods) {

      ofstream writefile;
      writefile.open("itemList.txt");

      writefile << purchase_day << endl;
      writefile << purchase_date << endl;

      for (int i = 0; i < number_goods; i++) {
            writefile << goods_name[i] << endl;
            writefile << goods_code[i] << endl;
            writefile << showpoint << fixed << setprecision(2);
            writefile << goods_price[i] << " " << goods_weight[i];
            if (i != (number_goods - 1))
                  writefile << endl;
      }
      writefile.close();
}


//7. Function addItemList --> add new items into the existing list
//    Hint: P7 should be a reference parameter as the number of items should
be updated
void addItemList(string goods_name[], string goods_code[], double
goods_price[], double goods_weight[], int& number_goods) {

      char select[20];
      int number = 0, alphabet = 0, unknown = 0;
      bool not_correct = true;

      //8. This part is to confirm whether user want to key in new item or
not.
      //    If yes, user will need to enter 1. Else, user will need to enter
2.
      //    An evaluation is carried out to ensure user only key in 1 or 2.
      //    However, the current evaluation is only able to check on
numbers.
      //    Additional marks will be awarded to those who are able to check
on the input other than numbers such as "abc" or "1abc".
      //    Tips: isdigit and stoi can be used. If you are using stoi with
dev c++, you may face c++ 11 problem and please
      //            refer to
https://stackoverflow.com/questions/13613295/how-can-i-compile-c11-code-
with-orwell-dev-c

      do {
        cout << "Add new item? (1-yes, 2-no): ";
        cin.getline(select, 20);

        for (int i = 0; i < strlen(select); i++) {

            if (isdigit(select[i]))
                number++;
            else if (isalpha(select[i]))
                alphabet++;
            else
                unknown++;
        }

        for (int i = 0; i < strlen(select); i++) {
            if ((number == 1) && (alphabet == 0) && (unknown == 0)) {
                switch (select[0]) {
                case '1':
                    not_correct = false;
                    break;
                case '2':
                    not_correct = false;
                    break;
                default:
                    number = 0;
                    alphabet = 0;
                    unknown = 0;
                    not_correct = true;
                }
            }
            else {
                number = 0;
                alphabet = 0;
                unknown = 0;
                not_correct = true;
            }
        }
    } while (not_correct);

    if (select[0] == '1') {
        //9. User will key in the details of the new items
        //   The COUT of add item for this module is shown in the A2
doc
        //   Marks will not be awarded if your COUT different from the
sample COUT
        for (int i = 0; i <= number_goods; i++) {
            if (i == number_goods) {
                cout << "New item's name: ";
                getline(cin, goods_name[number_goods]);
                cout << "New item's code: ";
                getline(cin, goods_code[number_goods]);
                cout << "Price of new item per KG (RM): ";
                cin >> goods_price[number_goods];
                cout << "Purchasing Weight (KG): ";
                cin >> goods_weight[number_goods];
            }
        }
        number_goods++;
    }
}

//10. Function modifyItemList --> modify the details of any existing items
in the list
void modifyItemList(string goods_name[], string goods_code[], double
goods_price[], double goods_weight[], int& number_goods) {
    int found_code = 0, select = 0, index_found = 0;
    string modify_code;

    //Evaluation is not required in this function
    do {
        cout << "Modify item? (1-yes, 2-no): ";
        cin >> select;

    } while (select <= 0 || select >= 3);

    if (select == 1) {
        //11. User will key in the item to be modified through item
code and find its match from the existing list
        //    If item is detected, all the details are to be keyed in
        //    Else, it will have to cout "Item is not available." as
shown in the A2 doc
        //    The COUT of modification for this module is shown in the
A2 doc
        //    Marks will not be awarded if your COUT different from the
sample COUT

        cin.ignore();
        cout << "Key in item code: ";
        getline(cin, modify_code);

        for (int i = 0; i < number_goods; i++) {
            if (goods_code[i] == modify_code) {
                found_code = 1;
                index_found = i;
            }
        }

        if (found_code == 1) {
            cout << "Item found!" << endl;
            cout << "New item's name: ";
            getline(cin, goods_name[index_found]);
```

```
                  cout << "New item's code: ";
                  getline(cin, goods_code[index_found]);
                  cout << "Price per KG (RM): ";
                  cin >> goods_price[index_found];
                  cout << "Purchasing Weight (KG): ";
                  cin >> goods_weight[index_found];
            }
            else
                  cout << "Item is not available." << endl;
            found_code = 0;
      }
}

//12. Function displayItemList --> display the details of all the items in
the list
//    The COUT of display items is shown in the A2 doc
//    Marks will not be awarded if your COUT different from the sample COUT
void displayItemList(string goods_name[], string goods_code[], double
goods_price[], double goods_weight[], int& number_goods) {
      for (int i = 0; i < number_goods; i++) {
            cout << "Item " << i + 1 << endl;
            cout << "Name: " << goods_name[i] << endl;
            cout << "Code: " << goods_code[i] << endl;
            cout << showpoint << fixed << setprecision(2);
            cout << "Price per KG: RM" << goods_price[i] << endl;
            cout << "Purchasing Weight: " << goods_weight[i] << "KG" <<
endl << endl;
      }
}

//13. Function printInvoice --> print out an invoice named "invoice.txt"
//    The output of the invoice is shown in the A2 doc
//    Marks will not be awarded if the output of your invoice is different
from the sample invoice
void printInvoice(string& purchase_day, string& purchase_date,
      string goods_name[], string goods_code[], double goods_price[],
double goods_weight[], int& number_goods) {

      double total = 0;

      ofstream writefile;
      writefile.open("invoice.txt");

      writefile << "Date: " << purchase_date << endl;
      writefile << "Day: " << purchase_day << endl;
      writefile << "List of Items:" << endl;
      for (int i = 0; i < number_goods; i++) {
            writefile << i + 1 << ". " << goods_name[i] << " = " <<
showpoint << fixed << setprecision(2) << goods_weight[i] << "KG"
```

```
                  << " * RM" << goods_price[i] << " = RM" <<
(goods_weight[i] * goods_price[i]) << endl;
            total += (goods_weight[i] * goods_price[i]);
      }
      writefile << "Grand Total of the bill = RM" << total;
      writefile.close();
}

//14. Function calculateGrandTotal --> calculate the grand total of the
items purchase and RETURN the grand total to the calling function
double calculateGrandTotal(double goods_price[], double goods_weight[],
int& number_goods) {
      double total = 0;
      for (int i = 0; i < number_goods; i++)
            total += (goods_weight[i] * goods_price[i]);
      return total;
}
```