COMPSCI 101

Principles of Programming

Lecture 21 – Maintaining a text file of information

Learning outcomes

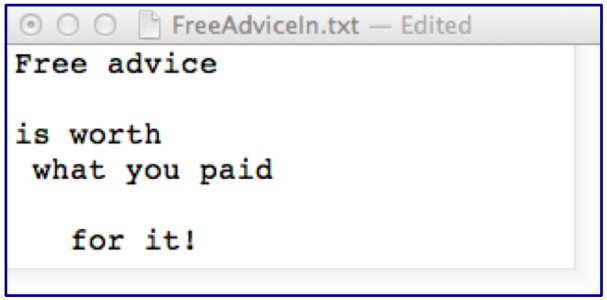
At the end of this lecture, students should be able to:

- read the contents of a text file into a list
- obtain, process, and update the data from the file
- use the split function to divide a string into different parts
- write the updated content back to a text file

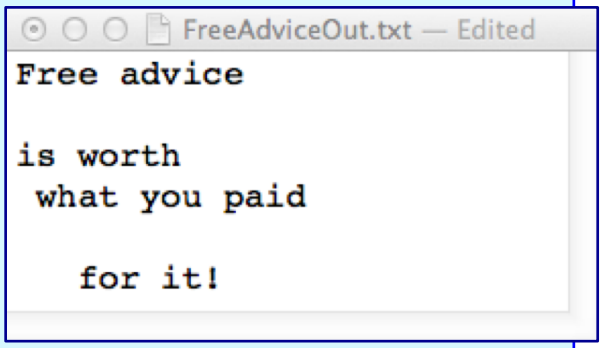
Recap

From lecture 20

- a file can be opened and closed
- data can be written to a file
- data can be read from a file




```
def copy_file(filename_in, filename_out):
    input_file = open(filename_in, "r")
    output_file = open(filename_out, "w")
    content = input_file.read()
    output_file.write(content)
    input_file.close()
    output_file.close()
    if len(content) < 8:
        return content
    return content[:4] + content[-4:]

def main():
    input_f = "FreeAdviceIn.txt"
    output_f = "Slide26FreeAdviceOut.txt"
    first_last_chars = copy_file(input_f, output_f)
    print(first_last_chars)
main()
```




```
Free it!
```

Remember the `split()` function - example

```
def main():
  words = "The budget was unlimited,  but I exceeded it  "

  word_list = words.split()

  print(words)
  print(word_list)

main()
```

```
The budget was unlimited,    but I exceeded it
['The', 'budget', 'was', 'unlimited,', 'but', 'I', 'exceeded', 'it']
```

Note about split(). If no separator is defined, whitespace is the separator.

Remember the `split()` function - example

The `split()` function separates a single string into a list of the parts of the string using the separator defined. The desired separator is passed to the `split()` function as a parameter, e.g.,

```
def main():
    words = "The,budget,was,unlimited   ,but,I,   exceeded,it  "
    word_list = words .split(",")

    print("1.", words)
    print("2.", word_list)

main()
```

```
The,budget,was,unlimited   ,but,I,   exceeded,it
['The', 'budget', 'was', 'unlimited   ', 'but', 'I', '   exceeded',
                                                        'it   ']
```

Online shopping example

A file, stock.txt, contains information about the items on sale in a simple online shopping system.

- Each line contains the information about one item on sale. The line is made up of the barcode, a description, the price and the quantity (number currently on stock).

During a shopping scenario users can:

- Place an item in the shopping cart,
- Update the item when it is bought,
- Check out the shopping cart, which results in the bill being generated,
- Save the file of stock.

Note that items are identified by their item code, e.g., 'bc###'.

stock.txt — Edited

```
bc001,Fresh toast bread white (700g),3.99,20
bc002,Low-fat milk (2 litre),4.8,10
bc003,V-energy drink,2.75,9
bc004,Fresh garlic (450g),1.98,4
bc005,Coca-Cola (300 ml),2.5,10
bc006,Pineapple,3.6,6
bc007,Mango,1.89,4
bc008,Snickers chocolate bar,1.8,20
bc009,Broccoli,1.47,11
bc010,Washed Potato (2.5kg),2.98,7
bc011,Cat food / Treats,2.75,15
bc012,pizza,6.54,4
bc013,pesto,9.44,2
bc014,Champagne,15.65,1
```

```
def main():
    items_list = load_stock("stock.txt")
    cart_list = []
    selection = 1

    while selection > 0:
        selection = get_menu_selection()
        if selection == 1:
            print_list(items_list)
        elif selection == 2:
            code_num = input("   Enter item code number: ")
            barcode = get_code_string(code_num)
            index = find_item_index(items_list, barcode)
            if index > -1:
                user_item = items_list[index]
                print("   Added to cart:", user_item)
                cart_list.append(user_item)
                update_quantity(items_list, index, -1)
            else:
                print("   This item does not exist.")
        elif selection == 3:
            print_list(cart_list)
```

```
    elif selection == 4:
        print_list(cart_list)
        cost = get_total(cart_list)
        print("  Total cost", "$" + str(cost))
        print("  -------------------------------")
        print("  -------------------------------")

    save_stock("stock2.txt", items_list)
```

The GoShopping.py program

```
def main():
        ....
def get_menu_selection():
    print()
    print("1. Display stock")
    print("2. Add item")
    print("3. Display cart")
    print("4. Check out shopping cart")
    print("0. Exit")
    return int(input("   Enter selection: "))
#----------------------------
#Print the list of items
#----------------------------
def print_list(a_list):
    for item in a_list:
        print("  ", item)
#----------------------------
#Create a code, e.g., bc003
#----------------------------
def get_code_string(num_str):
    code = "bc0" + num_str
    if int(num_str) < 10:
        code = "bc00" + num_str
    return code
```

The GoShopping program – Three helper functions

Assumption: the user never buys an item for which there is 0 quantity in stock.

Online shopping - stock.txt file

items_list →

0	'bc001,Fresh toast bread white (700g),3.99,20'
1	'bc002,Low-fat milk (2 litre),4.8,10 '
2	'bc003,V-energy drink,2.75,9'
3	'bc004,Fresh garlic (450g),1.98,4'
4	'bc005,Coca-Cola (300 ml),2.5,10'
5	'bc006,Pineapple,3.6,6'
...	'bc007,Mango,1.89,4'
	'bc008,Snickers chocolate bar,1.8,20'
	'bc009,Broccoli,1.47,11'
	'bc010,Washed Potato (2.5kg),2.98,7'
	'bc011,Cat food / Treats,2.75,15'
	'bc012,pizza,6.54,4'
	'bc013,pesto,9.44,2'
	'bc014,Champagne,15.65,1'

stock.txt — Edited

```
bc001,Fresh toast bread white (700g),3.99,20
bc002,Low-fat milk (2 litre),4.8,10
bc003,V-energy drink,2.75,9
bc004,Fresh garlic (450g),1.98,4
bc005,Coca-Cola (300 ml),2.5,10
bc006,Pineapple,3.6,6
bc007,Mango,1.89,4
bc008,Snickers chocolate bar,1.8,20
bc009,Broccoli,1.47,11
bc010,Washed Potato (2.5kg),2.98,7
bc011,Cat food / Treats,2.75,15
bc012,pizza,6.54,4
bc013,pesto,9.44,2
bc014,Champagne,15.65,1
```

From the text file into a list of strings.

Online shopping – load the stock into a list

The following slides all use the stock.txt file (see below).

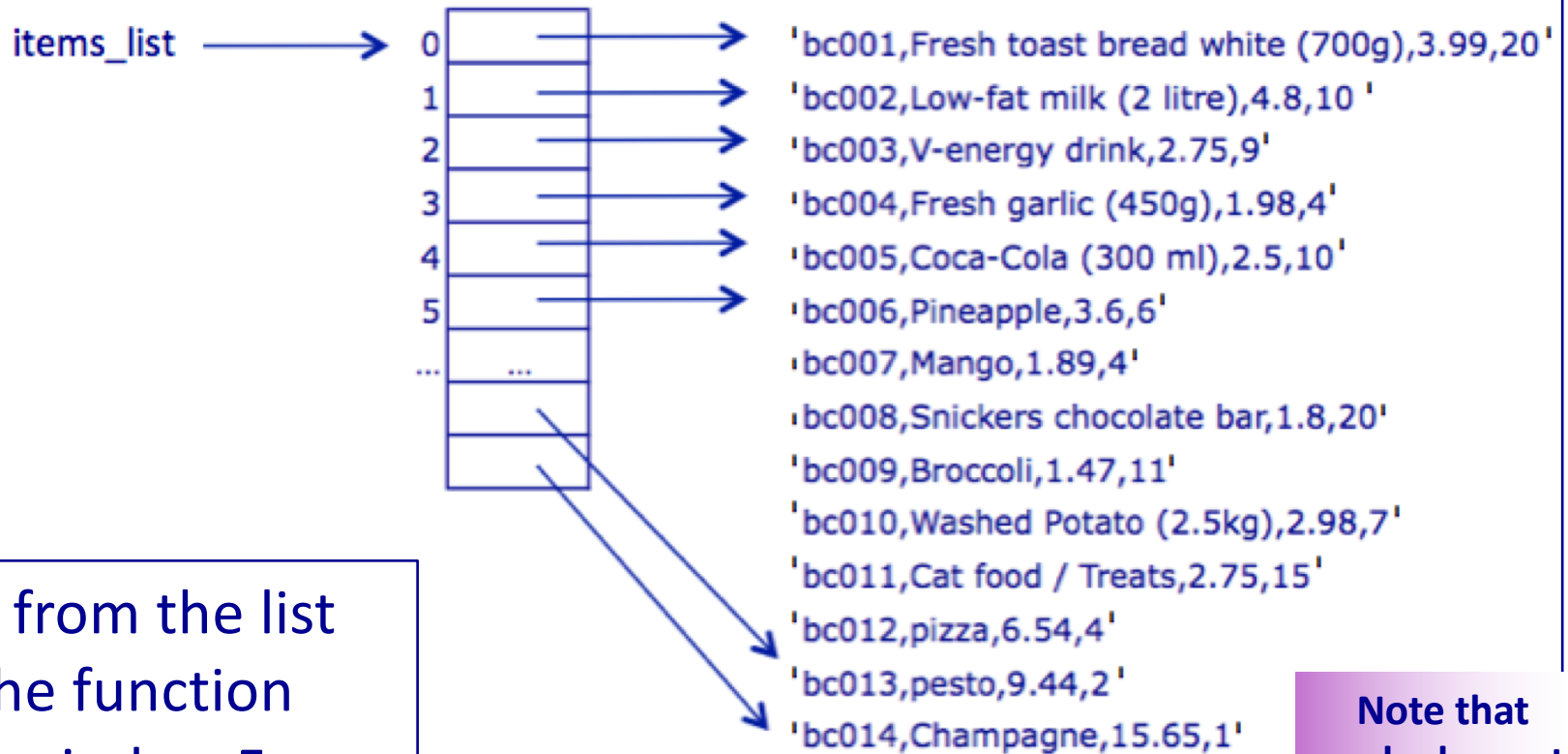
Read in contents of the stock file, and break up the contents of the file into a list of item records. Each list item is a single line (a string) from the stock.txt file. (In the file each line defining an item is separated from the next item by a newline character, "\n").

```
def load_stock(filename):




def main():
    items_list = load_stock("stock.txt")

main()
```

stock.txt — Edited

```
bc001,Fresh toast bread white (700g),3.99,20
bc002,Low-fat milk (2 litre),4.8,10
bc003,V-energy drink,2.75,9
bc004,Fresh garlic (450g),1.98,4
bc005,Coca-Cola (300 ml),2.5,10
bc006,Pineapple,3.6,6
bc007,Mango,1.89,4
bc008,Snickers chocolate bar,1.8,20
bc009,Broccoli,1.47,11
bc010,Washed Potato (2.5kg),2.98,7
bc011,Cat food / Treats,2.75,15
bc012,pizza,6.54,4
bc013,pesto,9.44,2
bc014,Champagne,15.65,1
```

Online shopping – find an item



Find an item from the list of strings. The function will return the index. For example find **bc003** returns the index 2 because this item is in index 2 of the list. Find **bc023** returns the index -1 because this item does not exist in the list.

Note that each element is a single string.

Online shopping – find an item

The find_item_index() function looks through the list of items to check whether the given code (e.g., 'bc001', 'bc002') exists. **Returns the index** if found, -1 if not found.

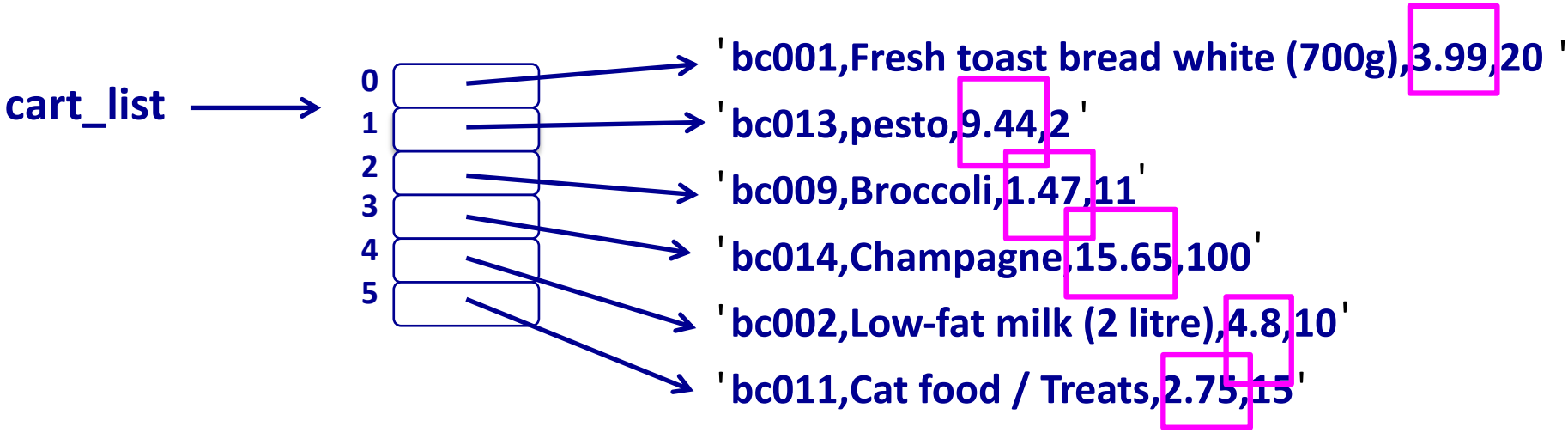
```
def find_item_index(items_list, code):




def main():
    code_num = input("   Enter item code number: ")
    barcode = get_code_string(code_num)
    index = find_item_index(items_list, barcode)
    if index != -1:
        print("   Added to cart:", items_list[index])
    else:
        print("   This item does not exist.")
main()
```

```
Enter item code: 3
Added to cart: bc003,V-energy drink,2.75,9
```

```
Enter item code: 23
This item does not exist.
```

Online shopping – total cost

To get the total cost of the list of items in the cart we need to sum the individual cost of each item.



Online shopping – total cost

To get the cost from one item (a string), we need to split the single string into a list of strings and obtain the information at position 2 in the list. The information needs to be converted into a float before it can be added to the total. For example,

"bc001,Fresh toast bread white (700g),3.99,20"

← One string

split_line →

0	‘bc001’
1	‘Fresh toast bread white (700g)’
2	‘3.99’
3	‘20’

← A list of strings

"3.99"

← A string

3.99

← A float

Online shopping – total cost

Complete the `get_total()` function.

```
def get_total(cart_list):


def main():
    ...
    elif selection == 4:
        print_list(cart_list)
        cost = get_total(cart_list)
        print("   Total cost", "$" + str(cost))

main()
```

Note: when each element of the cart_list is split into a list of its parts (comma separator), the cost of the item is in position 2 of the list.

```
bc006,Pineapple,3.6,6
bc014,Champagne,15.65,5
bc005,Coca-Cola (300 ml),2.5,10

Total cost $21.75
```

line shopping – update the quantity

To update the quantity of an item (a string), we need to add/subtract to/from the information at position 3 of the string. The information needs to be converted into an int, the amount added, and, the changed string needs to be assigned to the correct index of the item_list. E.g., the code:

```
energy_index = 2
update_quantity(items_list, energy_index , 5)
```

changes:

```
"bc003,V-energy drink,2.75,9"
```

into:

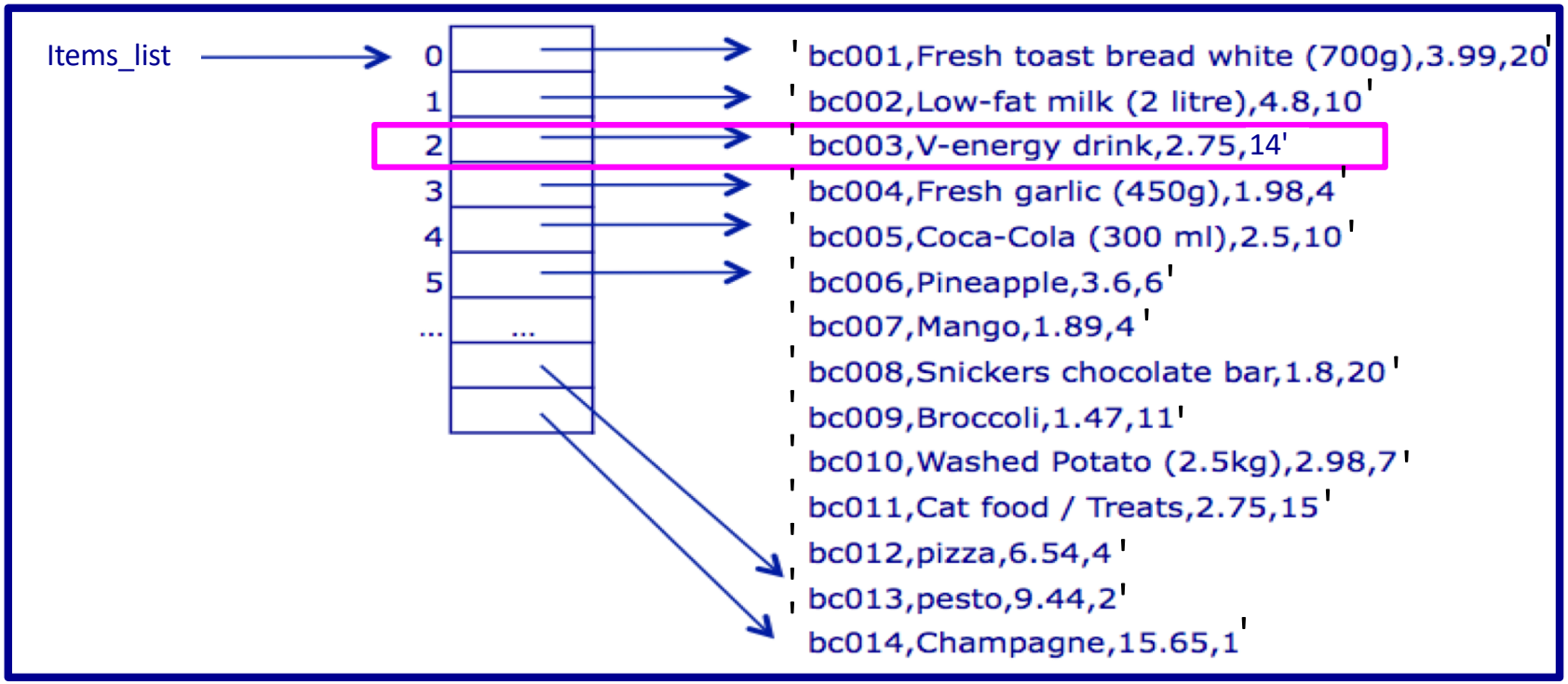
```
"bc003,V-energy drink,2.75,14"
```

line shopping – update the quantity

Finally, the changed string:

```
"bc003,V-energy drink,2.75,14"
```

needs to be assigned to the correct index of the item_list.



Online shopping – updating the quantity of an item on the stock list of items

When an item is added to (or removed from the shopping cart), its quantity value needs to be updated in the list of stock, items_list, e.g., for the following item:

```
'bc003,V-energy drink,2.75,9'
```

9 is the quantity, i.e, how many of this item are currently in stock.

When item, 'bc003' is bought (added to the shopping cart), the quantity for that item will decrease by one.

Note: in this program we are assuming that the user only buys an item if there is at least one of the item in stock.

Online shopping – updating the quantity of an item on the stock list of items

Complete the update_quantity() function.

Note: Assume the quantity will never be less than 0.

```
def update_quantity(items_list, index, update_amt):



def main():
    ...
    elif selection == 2:
            ...
            index = find_item_index(items_list, barcode)
            if index > -1:
                user_item = items_list[index]
                cart_list.append(user_item)
                update_quantity(items_list, index, -1)
              ...
main()
```

Online shopping – write the list to a file

Write the list of items to the filename file. Each item in the list is written on a new line in the file.

```
def save_stock(filename, items_list):




def main():
    items_list = = load_stock("stock.txt")
    …
    save_stock("stock2.txt", items_list)

main()
```

stock2.txt — Edited

```
bc001,Fresh toast bread white (700g),3.99,20
bc002,Low-fat milk (2 litre),4.8,10
bc003,V-energy drink,2.75,8
bc004,Fresh garlic (450g),1.98,3
bc005,Coca-Cola (300 ml),2.5,10
bc006,Pineapple,3.6,6
bc007,Mango,1.89,1
bc008,Snickers chocolate bar,1.8,20
bc009,Broccoli,1.47,11
bc010,Washed Potato (2.5kg),2.98,7
bc011,Cat food / Treats,2.75,15
bc012,pizza,6.54,4
bc013,pesto,9.44,2
bc014,Champagne,15.65,1
```

Summary

In a Python program:

- the contents of a file can be opened and read into a list
- data from a file can be obtained, processed, and updated
- the split function can be used to divide a string into different parts

Examples of Python features used in this lecture

```
def update_quantity(items_list, index, update_amt):
    item_string = items_list[index]
    item_parts = item_string.split(",")

    quantity = int(item_parts[3])
    quantity = quantity + update_amt

    quantity = max(quantity, 0)

    updated_str = ""
    for pos in range(len(item_parts) - 1):
        updated_str = updated_str + item_parts[pos] + ","

    updated_str = updated_str + str(quantity)

    items_list[index] = updated_str
```