**TRANSPORT AND TELECOMMUNICATION INSTITUTE**



Bachelor of Natural Sciences in Computer Science

Course Work

Made by:

Arturs Gabetovs

Student code: 61908

Group: 4902BDA

Considered by:

Jānis Pekša

Riga

2019

**Content**

[Introduction 3](#_Toc514333339)

[Task of the course work 3](#_Toc514333340)

[The volume of analysis and definition of tasks 3](#_Toc514333341)

[Initial design 3](#_Toc514333342)

[Flowcharts 4](#_Toc514333344)

[Testing 4](#_Toc514333345)

[Conclusion 5](#_Toc514333346)

[From the literature and Internet list sources. 9](#_Toc514333347)

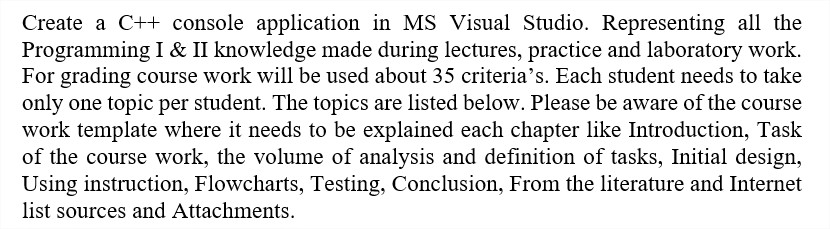
[Attachments 9](#_Toc514333348)

## Introduction

Hello, my Name is Arturs G. here I build it the program with few functions and checking if the user did write something wrong, this program will ask the user to write in the correct way. This program was optimized with the way I found and is as an advantage I took a little step forward and try to go to classes, just little step, but it’s done and works cool, which helps to view my code clearly and look cleaner.

To build this, it’s taken me a lot of time, to find the right answer to my question and my job is done with the way you will see below.

## Task of the course work



My theme of the Course Work:



## The volume of analysis and definition of tasks

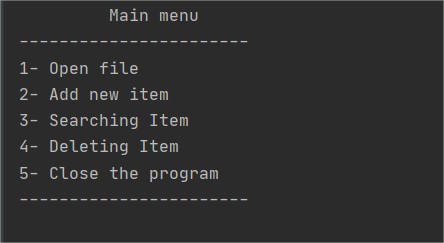
Firstly, I tried to understand my Theme of Course work by searching information in the internet and how usually stored item in the warehouses.

Secondly the way how to build the program and what my program should consist.

I highlighted 5 points what my program should have:

* Open File.
* Adding new Item to the file.
* Searching Item on the file.
* Deleting item from the file and make sure no other item will be deleted from the list.

## Initial design

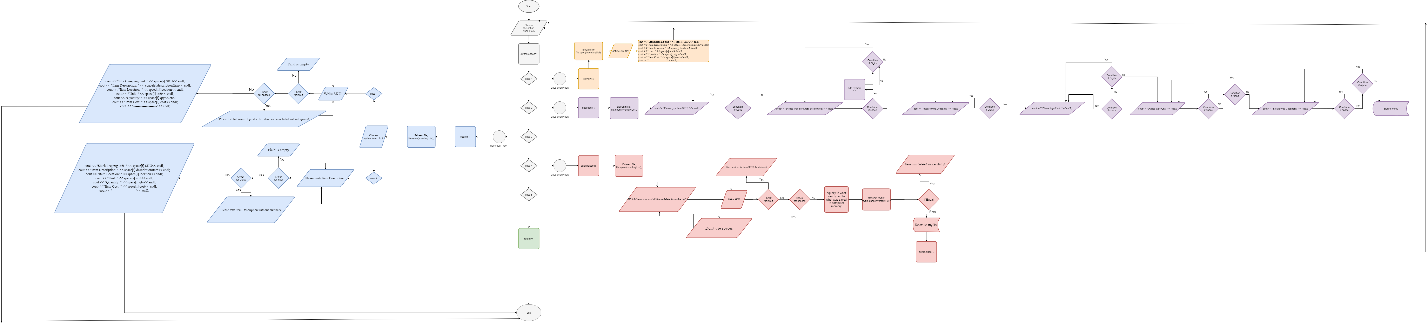


## Using instruction

After running the program, user will need only press the number which function he will like to do.

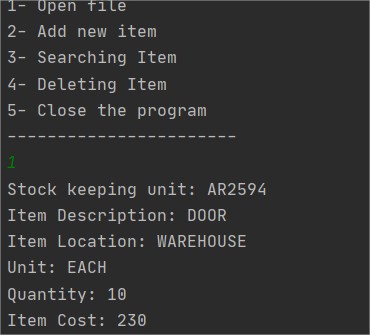
## Flowcharts

My flowchart in good resolution and here is explain how all my program is working I will also include my flowchart picture for better overview in course work.

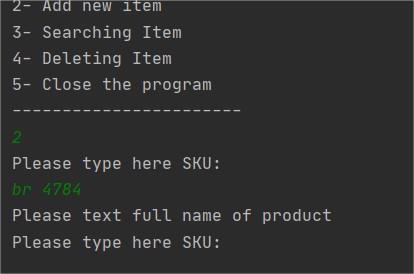


## Testing

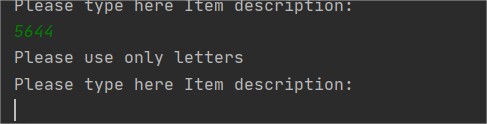
Test Nr. 1 Open List.



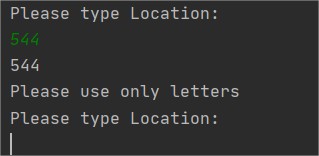
Test Nr. 2 Add Item to the List with all checking.



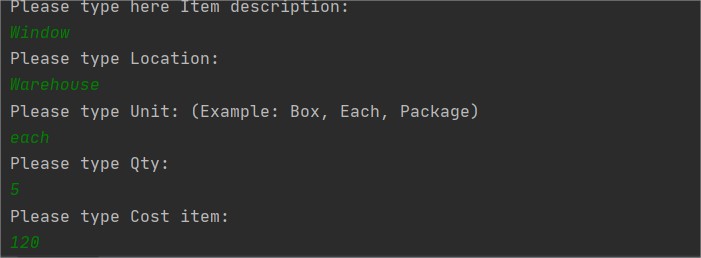
## Test Nr. 2-1 checking Item Description shouldn’t consist numbers.



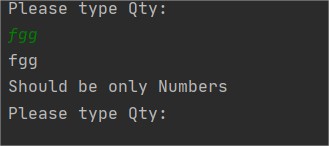
Test Nr. 2-2 Location.



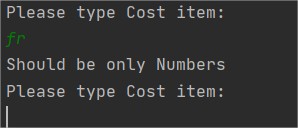
Test Nr. 2-3 Unit.



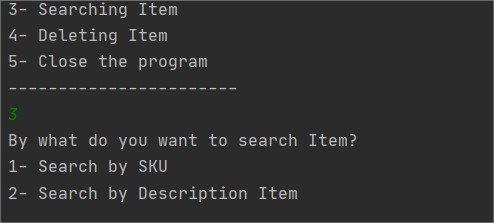
Test Nr. 2-4 Quantity.



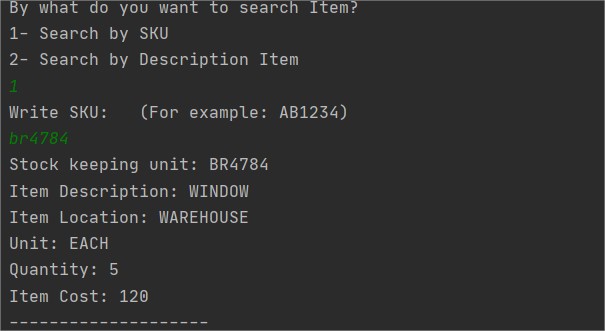
Test Nr. 2-5 Cost.



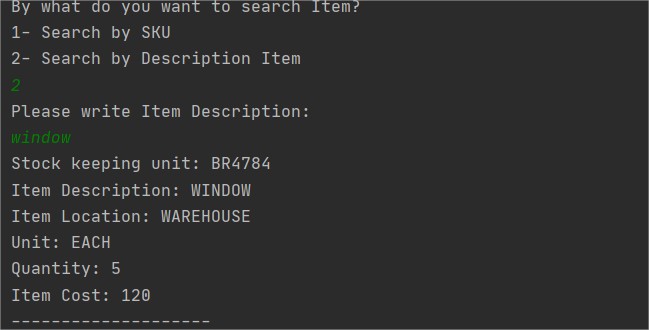
Test Nr. 3



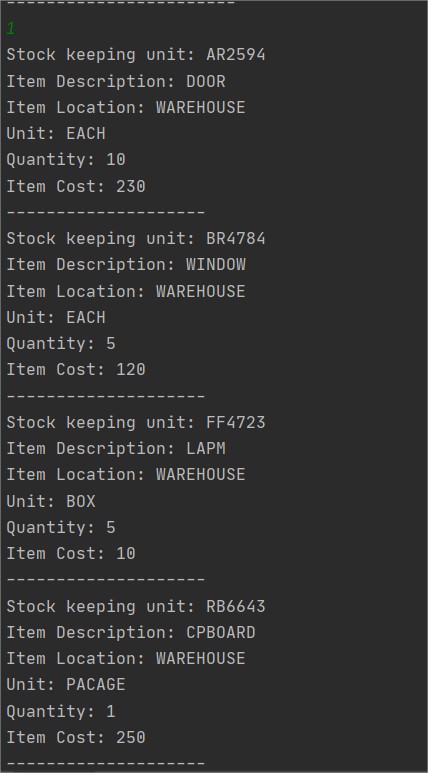
Test Nr. 3-1 Search by Stock keep unit.



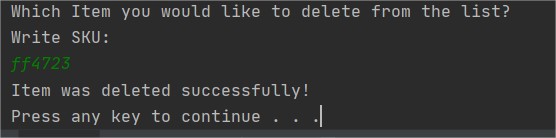
Test Nr. 3-2 Search by Item Description.



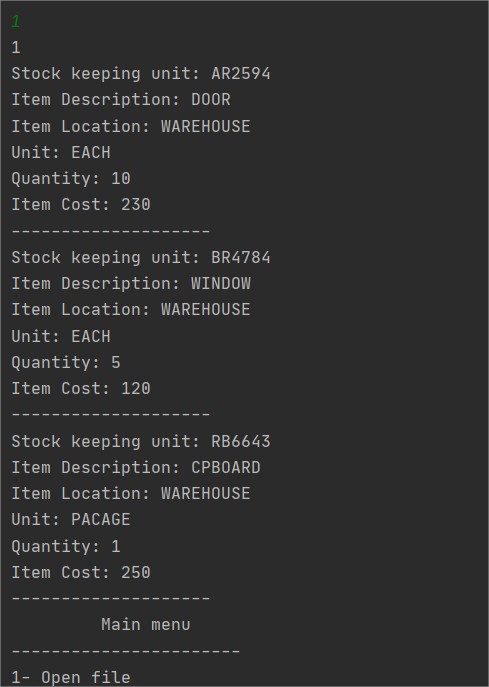
Test Nr. 4 Showing the List with existing Items.



Test Nr. 4-1 We are deleting Item FF4723



Test Nr. 4-2 Checking the List again after removing the Item.



Test Nr. 5 Closing the program.

## 

## Conclusion

It this Course work I tried to use all my knowledge from 2 semesters, there was using a couple of libraries which wasn’t used before and + new way how to sort all function which we didn’t pass yet. This work takes pretty a lot of time, to the researcher the way, how one or other functions will be work, and some conclusion was coming from the way “I don’t know how, but it works”.

I understand in my job I didn’t provide a lot of function and checking to make my work unbreakable, there is still weak places must-have. With this course work possible to upgrade to another level with all checking and optimizing the lines, because I believe there is still a lot of things to learn and this work could be good practice in the future.

## From the literature and Internet list sources

<https://www.w3schools.com/cpp/default.asp> (using web site, for remind some topics)

<https://stackoverflow.com/> (using a forum with a lot of issues and how to solve them)

<http://www.cplusplus.com/reference/fstream/> (for find the way how properly use “fstream” library and other function and checking)

## Attachments

Also, will be included Flowchart.png and invenoryList.txt file with all element.

## I build it my code with classes

main.cpp

#include "case1.h"  
  
using namespace std;  
  
int main() {  
  
 string option;  
 int option1;  
  
  
  
 cout << "\t Main menu" << endl;  
 cout << "-----------------------" << endl;  
 cout << "1- Open file" << endl;  
 cout << "2- Add new item" << endl;  
 cout << "3- Searching Item" << endl;  
 cout << "4- Deleting Item" << endl;  
 cout << "5- Close the program" << endl;  
 cout << "-----------------------" << endl;  
  
 getline(cin, option);  
 istringstream(option) >> option1;  
  
 switch (option1) {  
 case 1: {  
 userTypo list; //Open List  
 list.listView();  
 return main();  
  
 }break;  
 case 2: {  
 userTypo userText;  
 userText.questions(); // Adding the Item  
 return main(); // Going back to the Main menu  
 }  
 case 3:{  
 userTypo search; //Searching  
 search.searching();  
 return 0;  
 }  
  
 case 4:{  
 userTypo del; //Deleting Item  
 del.deletingItem();  
  
 system("pause");  
 return main();  
 }  
 case 5:{  
 return 0; //Closing Program  
 }  
 }  
 }

case1.h sub-class: where I was writing all my functions

#ifndef UNTITLED1\_CASE1\_H  
#define UNTITLED1\_CASE1\_H  
#include <iostream>  
#include <fstream> //Library for working with files  
#include <string>  
#include<sstream> //Using this library for converting string to int  
  
using namespace std;  
  
string inventoryList = "InventoryList.txt";  
  
struct ItemList {  
 string SKU; //переменная = variable  
 string descriptionItem;  
 string location;  
 string unit;  
 string qty;  
 string cost;  
}space[50];  
  
class userTypo {  
public:  
  
 void listView(){ // Function Show list  
  
  
 string item;  
 ifstream file;  
 int i = 0;  
 file.open(inventoryList);  
  
 while(getline(file, item)){  
  
 istringstream view(item);  
 view >> space[i].SKU >> space[i].descriptionItem>> space[i].location>> space[i].unit>> space[i].qty >> space[i].cost;  
  
 cout << "Stock keeping unit: " << space[i].SKU<< endl;  
 cout << "Item Description: " << space[i].descriptionItem<< endl;  
 cout << "Item Location: " << space[i].location<< endl;  
 cout << "Unit: " << space[i].unit<< endl;  
 cout << "Quantity: " << space[i].qty<< endl;  
 cout << "Item Cost: " << space[i].cost<< endl;  
 cout << "--------------------"<< endl;  
 i++;  
 }  
 }  
  
  
 static void questions(){ // Function Question  
  
 ofstream file;  
 file.open(inventoryList, ofstream::app);  
  
 SKU: //Returning point  
  
 cout << "Please type here SKU: "<< endl;  
 getline(cin, space[0].SKU);  
  
 for( int i =0; i< space[0].SKU.length(); i++)  
 {  
 if (isspace(space[0].SKU[i])){ //checking for spaces !  
 cout<< "Please text full name of product"<< endl;  
 goto SKU; // coming back to the SKU !  
 }  
 space[0].SKU[i] = toupper(space[0].SKU[i]);  
 }  
  
 desItem:  
  
 cout << "Please type here Item description: "<< endl;  
 getline(cin, space[0].descriptionItem);  
  
 for( int i =0; i< space[0].descriptionItem.length(); i++){  
  
 if (isspace(space[0].descriptionItem[i])){  
 space[0].descriptionItem[i] = '\_';  
 }  
 if (isdigit(space[0].descriptionItem[i])){  
 cout<< "Please use only letters" << endl;  
 goto desItem;  
 }  
 space[0].descriptionItem[i] = toupper(space[0].descriptionItem[i]);  
 }  
  
 loc:  
 cout << "Please type Location: "<< endl;  
 getline(cin, space[0].location);  
  
 for(int i=0; i< space[0].location.length(); i++){  
 if (isdigit(space[0].location[i])){  
 cout<< "Please use only letters" << endl;  
 goto loc;  
 }  
 space[0].location[i] = toupper(space[0].location[i]);  
 }  
  
 unit:  
 cout << "Please type Unit: (Example: Box, Each, Package)"<< endl;  
 getline(cin, space[0].unit);  
  
 for(int i=0; i< space[0].unit.length(); i++){  
  
 if (space[0].unit.empty() || isdigit(space[0].unit[i])){  
 cout<< "Please use only letters" << endl;  
 goto unit;  
 }  
 space[0].unit[i] = toupper(space[0].unit[i]);  
 }  
  
 qty:  
 cout << "Please type Qty: "<< endl;  
 getline(cin, space[0].qty);  
  
 for( int i =0; i< space[0].qty.length(); i++) {  
 if (isspace(space[0].qty[i])|| isalpha(space[0].qty[i])) {  
 cout << "Should be only Numbers" << endl;  
 goto qty;  
 }  
 }  
  
 cost:  
 cout << "Please type Cost item: "<< endl;  
 getline(cin, space[0].cost);  
  
 for( int i =0; i< space[0].cost.length(); i++) {  
 if (isspace(space[0].cost[i])|| isalpha(space[0].cost[i])) {  
 cout << "Should be only Numbers" << endl;  
 goto cost;  
 }  
 }  
 system("cls");  
  
 file << space[0].SKU<<" ";  
 file << space[0].descriptionItem<<" ";  
 file << space[0].location<<" ";  
 file << space[0].unit<<" ";  
 file << space[0].qty<<" ";  
 file << space[0].cost<<" ";  
 file << endl;  
 file.close();  
 }  
 static void searching(){ //Search  
  
 ifstream file;  
 string look;  
 file.open(inventoryList);  
 int i = 0;  
 while(getline(file, look)){  
  
 istringstream check(look);  
 check >> space[i].SKU >> space[i].descriptionItem>> space[i].location>> space[i].unit>> space[i].qty >> space[i].cost;;  
 i++;  
 }  
  
 int search1;  
 string search;  
  
 searchmenu:  
  
 cout << "By what do you want to search Item?" << endl;  
 cout << "1- Search by SKU" << endl;  
 cout << "2- Search by Description Item" << endl;  
 getline (cin,search);  
  
 istringstream (search)>> search1;  
 switch(search1){  
 case 1:{  
 string searchSKU;  
 point1:  
 cout << "Write SKU: (For example: AB1234)"<< endl;  
 getline(cin, searchSKU);  
  
 if (searchSKU.empty()){  
 cout << "Field is empty"<< endl;  
 system("cls");  
 goto searchmenu;  
 }  
 for( int i = 0; i< searchSKU.length(); i++)  
 {  
 if (isspace(searchSKU[i])){ //checking for spaces !  
 cout<< "Please text full name of product Nr. also can be included without spaces!"<< endl;  
 goto point1; // coming back to the SKU !  
 }  
 searchSKU[i] = toupper(searchSKU[i]);  
 }  
 for(int i = 0; i< 50; i++){  
 if(searchSKU == space[i].SKU){  
  
 cout << "Stock keeping unit: " << space[i].SKU<< endl;  
 cout << "Item Description: " << space[i].descriptionItem<< endl;  
 cout << "Item Location: " << space[i].location<< endl;  
 cout << "Unit: " << space[i].unit<< endl;  
 cout << "Quantity: " << space[i].qty<< endl;  
 cout << "Item Cost: " << space[i].cost<< endl;  
 cout << "--------------------"<< endl;  
 }  
 }  
  
 }break;  
 case 2:{  
  
 string searchDescriptionItem;  
  
 searchDesItem:  
 point2:  
 cout << "Please write Item Description: " << endl;  
 getline(cin, searchDescriptionItem);  
  
 if (searchDescriptionItem.empty()){  
 cout << "Field is empty"<< endl;  
 system("cls");  
 goto searchDesItem;  
  
 }  
 for( int i = 0; i< searchDescriptionItem.length(); i++)  
 {  
 if (isdigit(searchDescriptionItem[i])){ //checking for spaces !  
 cout<< "Please text Item Description without numbers"<< endl;  
 goto point2; // coming back to the SKU !  
 }  
 searchDescriptionItem[i] = toupper(searchDescriptionItem[i]);  
 }  
 for(int i = 0; i< 50; i++){  
 if(searchDescriptionItem == space[i].descriptionItem){ //checking if written descriptionItem are equal  
 // to Description Item inside the file  
 cout << "Stock keeping unit: " << space[i].SKU<< endl;  
 cout << "Item Description: " << space[i].descriptionItem<< endl;  
 cout << "Item Location: " << space[i].location<< endl;  
 cout << "Unit: " << space[i].unit<< endl;  
 cout << "Quantity: " << space[i].qty<< endl;  
 cout << "Item Cost: " << space[i].cost<< endl;  
 cout << "--------------------"<< endl;  
 }  
 }  
 }  
  
 }  
 }  
 static void deletingItem(){ // Function Deleting  
  
 ifstream file;  
 string look;  
 string deleteItem;  
 file.open(inventoryList);  
 int i = 0;  
 int y = 0;  
  
 while(getline(file, look)){ //Taking all information from file inventoryList  
  
 istringstream check(look);  
 check >> space[i].SKU >> space[i].descriptionItem>> space[i].location>> space[i].unit>> space[i].qty >> space[i].cost;;  
 i++;y++;  
 }  
 file.close();  
  
 cout << "Which Item you would like to delete from the list?" << endl;  
 delet:  
 cout << "Write SKU: " << endl;  
 getline(cin,deleteItem);  
  
 if(deleteItem.empty()){ // if User wrote nothing  
 cout << "You must write item SKU for delete it"<< endl;  
 goto delet;  
 }  
 for(int i = 0; i < deleteItem.length(); i++){ // taking length of deleteItem variable  
 if(isspace(deleteItem[i])){ // checking for spaces  
 cout <<"Don't use spaces"<<endl;  
 goto delet;  
 }  
 deleteItem[i] = toupper(deleteItem[i]);  
 }  
 ofstream myfile;  
 myfile.open(inventoryList);  
  
 for(int i = 0; i < y; ++i){  
  
 if(deleteItem == space[i].SKU){ // If equal deleting found item  
 cout << "Item was deleted successfully!" << endl;  
  
 }  
 else {  
 //Re-writing file with current result  
 myfile << space[i].SKU<<" ";  
 myfile << space[i].descriptionItem<<" ";  
 myfile << space[i].location<<" ";  
 myfile << space[i].unit<<" ";  
 myfile << space[i].qty<<" ";  
 myfile << space[i].cost<<" ";  
 myfile << endl;  
 }  
 }  
 myfile.close();  
 }  
  
};  
  
#endif //UNTITLED1\_CASE1\_H