− Титульный лист

− ВВЕДЕНИЕ

− 1. ПОСТАНОВКА ЗАДАЧИ

+ 2. ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ К ПРОГРАММЕ

+ 3. АЛГОРИТМ ПРОБЛЕМНОЙ ЧАСТИ

+ 4. ВЫБОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

± 5. ПРОГРАММНЫЙ КОД

(TODO: Неиспользуемые подпрограммы)

± 6. ОТЛАДКА И ТЕСТИРОВАНИЕ

− 7. АНАЛИЗ РЕЗУЛЬТАТОВ

Функциональные требования к программе

Программа считывает файл, записанный в формате CSV-таблицы, и преобразовывает каждую строку в структуру, описывающую один товар. В итоге получается массив структур. Далее программа выдаёт следующую информацию:

1. Наличие и стоимость изделия артикула Х (вводится с терминала);
2. Ассортиментный список детского трикотажа с указанием наименования, если известно, что артикул начинается с буквы W для дамского трикотажа, M – для мужского, C – для детского

Алгоритм проблемной части



Выбор программного обеспечения

Выбранная среда разработки – CLion.

Преимущества:

* Поддержка компиляции в WSL
* Статический анализатор кода с подсветкой синтаксиса
* Интеграция с Git/GitHub и другими VCS
* Человеческий дебагер
* Темная тема
* Поддержка копирования кода в Word с сохранением форматирования
* Поддержка экспорта кода в HTML
* Авто-рефактор кода
* Поддержка кучи плагинов

Программный код

main.cpp

#include <iostream>  
#include "module.h"  
  
using namespace std;  
  
  
int main() {  
 auto db = openDB("db.csv");  
  
 char code[CODE\_SIZE];  
 cout << "Введите артикул: " << endl;  
 cin >> code;  
 print\_by\_code(code, db);  
 print\_c\_code(db);  
  
 return 0;  
}

module.h

#ifndef LAB16\_\_\_MODULE\_H  
#define LAB16\_\_\_MODULE\_H  
  
#define CODE\_SIZE 4  
#define NAME\_SIZE 32  
#define DB\_SIZE 65535  
  
struct Item {  
 char code[CODE\_SIZE];  
 char name[NAME\_SIZE];  
 int amount;  
 int price;  
};  
  
struct DB {  
 int len;  
 Item\* table;  
};  
  
Item to\_item(const char \*str);  
DB openDB(const char \*f\_name);  
void print\_by\_code(const char \*str, DB data);  
void print\_c\_code(DB data);  
  
#endif //LAB16\_\_\_MODULE\_H

module.cpp

#include <iostream>  
#include <iomanip>  
#include <cstring>  
#include "module.h"  
  
using namespace std;  
  
Item to\_item(const char \*str) {  
 Item out;  
 int text\_index = 0;  
 memset(out.code, '\0', CODE\_SIZE);  
 for (; str[text\_index] != ',' && text\_index < CODE\_SIZE; text\_index++) {  
 out.code[text\_index] = str[text\_index];  
 }  
 memset(out.name, '\0', NAME\_SIZE);  
 int local\_index;  
 for (local\_index = 0, text\_index++;  
 str[text\_index] != ',' && local\_index < NAME\_SIZE; text\_index++, local\_index++) {  
 out.name[local\_index] = str[text\_index];  
 }  
 char amount\_str[10] = {'\0'};  
 for (local\_index = 0, text\_index++; str[text\_index] != ',' && local\_index < 10; text\_index++, local\_index++) {  
 amount\_str[local\_index] = str[text\_index];  
 }  
 out.amount = atoi(amount\_str);  
 char price\_str[10] = {'\0'};  
 for (local\_index = 0, text\_index++; str[text\_index] != ',' && local\_index < 10; text\_index++, local\_index++) {  
 price\_str[local\_index] = str[text\_index];  
 }  
 out.price = atoi(price\_str);  
 return out;  
}  
  
DB openDB(const char \*f\_name) {  
 FILE \*fp;  
 if ((fp = fopen(f\_name, "r")) == nullptr) {  
 puts("Невозможно открыть файл");  
 exit(-1);  
 }  
  
 int len = 1; // Последняя строка без '\n'  
 char buff;  
 char text[DB\_SIZE];  
 int text\_index = 0;  
 while ((buff = getc(fp)) != EOF) {  
 if (buff == '\n') len++;  
 text[text\_index++] = buff;  
 }  
 fclose(fp);  
  
 Item db[len];  
 char \*text\_ptr = text;  
 for (int i = 0; i < len; i++) {  
 db[i] = to\_item(text\_ptr);  
 text\_ptr = strchr(text\_ptr, '\n') + 1;  
 }  
 return DB{len, db};  
}  
  
void print\_by\_code(const char \*code, DB data) {  
 auto[len, db] = data;  
 cout << "┌" << string(CODE\_SIZE, '-') << "┬" << string(NAME\_SIZE, '-') << "┬" << string(10, '-') << "┬" << string(10, '-') << "┐" << endl;  
 cout << "|" << left << setw(CODE\_SIZE) << "Code" << "|" << setw(NAME\_SIZE) << "Name" << "|" << setw(10) << "Amount" << "|" << setw(10) << "Price" << "|" << endl;  
 cout << "├" << string(CODE\_SIZE, '-') << "┼" << string(NAME\_SIZE, '-') << "┼" << string(10, '-') << "┼" << string(10, '-') << "┤" << endl;  
 for (int i = 0; i < len; i++) {  
 if (strcmp(code, db[i].code) == 0) {  
 cout << "|" << left << setw(CODE\_SIZE) << db[i].code << "|" << setw(NAME\_SIZE) << db[i].name << "|" << right << setw(10) << db[i].amount << "|" << setw(10) << db[i].price << "|" << endl;  
 }  
 }  
 cout << "└" << string(CODE\_SIZE, '-') << "┴" << string(NAME\_SIZE, '-') << "┴" << string(10, '-') << "┴" << string(10, '-') << "┘" << endl;  
}  
  
void print\_c\_code(DB data) {  
 auto[len, db] = data;  
 cout << "Детская одежда" << endl;  
 cout << "┌" << string(CODE\_SIZE, '-') << "┬" << string(NAME\_SIZE, '-') << "┬" << string(10, '-') << "┬" << string(10, '-') << "┐" << endl;  
 cout << "|" << left << setw(CODE\_SIZE) << "Code" << "|" << setw(NAME\_SIZE) << "Name" << "|" << setw(10) << "Amount" << "|" << setw(10) << "Price" << "|" << endl;  
 cout << "├" << string(CODE\_SIZE, '-') << "┼" << string(NAME\_SIZE, '-') << "┼" << string(10, '-') << "┼" << string(10, '-') << "┤" << endl;  
 for (int i = 0; i < len; i++) {  
 if (strncmp("C", db[i].code, 1) == 0) {  
 cout << "|" << left << setw(CODE\_SIZE) << db[i].code << "|" << setw(NAME\_SIZE) << db[i].name << "|" << right << setw(10) << db[i].amount << "|" << setw(10) << db[i].price << "|" << endl;  
 }  
 }  
 cout << "└" << string(CODE\_SIZE, '-') << "┴" << string(NAME\_SIZE, '-') << "┴" << string(10, '-') << "┴" << string(10, '-') << "┘" << endl;  
}

ОТЛАДКА И ТЕСТИРОВАНИЕ

Исходная таблица

W01,Dress,100,1000  
W02,T-shirt,7,4015  
W03,Blouse,16,357  
W04,Sweater,16,635  
W05,Skirt,74,3038  
W06,Jeans,57,3198  
M01,Suite,50,3000  
M02,T-shirt,40,587  
M03,Sweater,21,4604  
M04,Tracksuit,35,1447  
M05,Jeans,33,1094  
C01,Hat,25,100  
C02,T-shirt,55,4447  
C03,Sweater,58,3910  
C04,Skirt,98,2082

Вывод программы

