**Міністерство Освіти І НАУКИ України**

**Національний університет "Львівська політехніка"**

**Інститут комп’ютерних наук та інформаційних технологій**

**Кафедра програмного забезпечення**



### ЗВІТ

До лабораторної роботи № 2

**На тему:***“Документування етапів проектування та кодування програми”*

**З дисципліни:** *“Вступ до інженерії програмного забезпечення”*

**Лектор:**

доцент Левус Є. В.

**Виконала:**

ст. гр. ПЗ-16

Гук А.М.

**Прийняв:**

асист. Самбір А. А.

« \_\_\_\_ » \_\_\_\_\_\_\_\_ 2022 р.

∑= \_\_\_\_ .

Львів – 2022

**Тема роботи:**документування етапів проектування та кодування програми.

**Мета роботи:**навчитися документувати основні результати етапів проектування та кодування найпростіших програм.

**Теоретичні відомості**

***2.*** *У чому полягає етап проектування для найпростішої програми?*

На етапі проектування необхідно описати, з яких частин складатиметься ПЗ й як воно буде працювати, тобто необхідно задати структуру й поведінку ПЗ. Отримані проектні рішення у подальшому переводяться в програмні коди обраною мовою програмування. Причому можливий автоматизований перехід від проектних рішень до кодів програми.

**18.** .*Які вимоги до запису коментарів у тексті програми?*

Рекомендується використовувати стиль однострічкових коментарів через //. Для коментування великого фрагменту коду у процесі відлагодження використовується /\* \*/.

**32.** *Що таке рефакторинг коду?*

Рефакторинг коду – один з типових процесів, що полягає у⎫ перетворенні програмного коду, зміні внутрішньої структури програмного забезпечення для полегшення розуміння коду і легшого внесення подальших змін без зміни зовнішньої поведінки самої системи.

**Постановка завдання**

**Частина I.**

У розробленій раніше програмі до лабораторної роботи з дисципліни «Основи програмування» внести зміни – привести її до модульної структури, де модуль – окрема функція-підпрограма. У якості таких функцій запрограмувати алгоритми зчитування та запису у файл, сортування, пошуку, редагування, видалення елементів та решта функцій згідно варіанту.

**Частина II.**

Сформувати пакет документів до розробленої раніше власної програми:

1. Схематичне зображення структур даних, які використовуються для збереження інформації;
2. Блок-схема алгоритмів – основної функції й двох окремих функцій-підпрограм (наприклад, сортування та редагування);
3. Текст програми з коментарями та оформлений згідно вище наведених рекомендацій щодо забезпечення читабельності й зрозумілості.

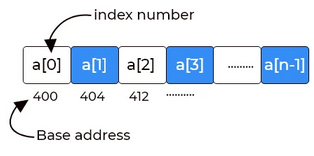
Для схематичного зображення структур даних, блок-схеми алгоритму використати редактор MS-Visio.

**Частина III.**

У редакторі MS-Visioрозробити зразки фігур, які були використані для схематичного зображення структур даних програм, як готові трафарети до використання. Сформувати свою бібліотеку фігур – окремий користувацький файл із використаними зразками.

**Отримані результати**

1. Схематичне зображення використаних структур даних:

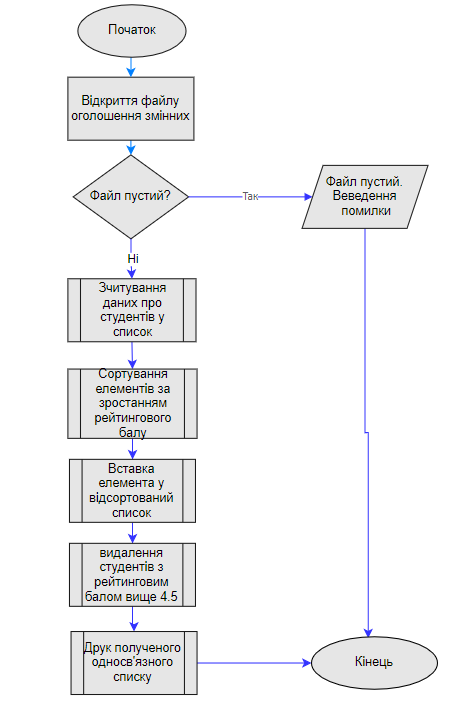


*Рис.1 Масив*

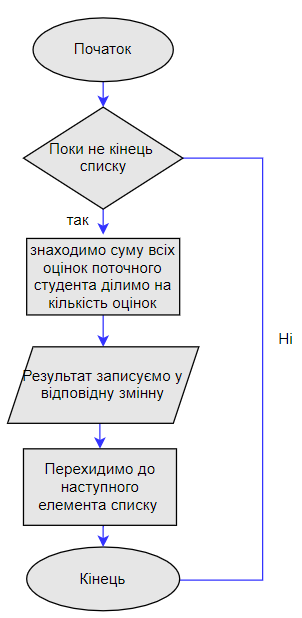
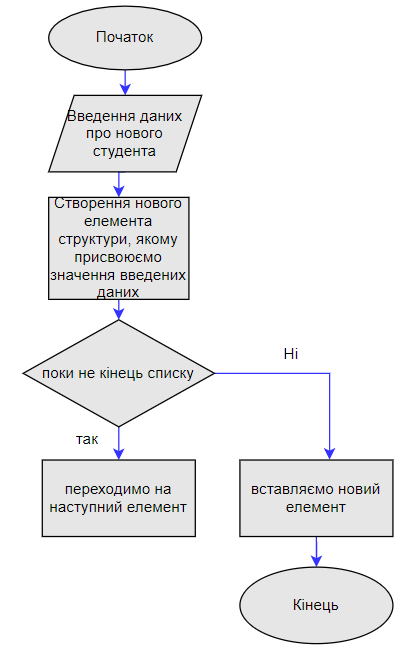


*Рис. 2 Однозв’язний список*

1. Блок-схеми алгоритмів:



*Рис. 3 Блок-схема головної програми*

* *

*Рис. 4,5 Блок-схеми алгоритму обрахунку рейтингового балу та алгоритму додавання нового вузла в список*

1. Сформований користувацький Visio-файл фігур:



1. Текст програми:

main.c

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include "functions.h"  
  
int main() {  
  
 students\* head = NULL;  
 printf("List of students\n");  
 readFromFile(&head);  
 printf("Enter surname, name, birthday and 5 marks of inserted element:\n");  
 insertAtEnd(head);  
 printf("\n");  
 findRatingMark(head);  
 bubbleSort(head);  
 deleteUpper4\_5(&head);  
 printf("Modificated list of students");  
 displayList(head);  
 return 0;  
}

Function.h

#ifndef LABORATORY\_10\_FUNCTIONS\_H  
#define LABORATORY\_10\_FUNCTIONS\_H  
#define CHSIZE 20  
#define MARKSIZE 5  
  
typedef struct students  
{  
 char name[CHSIZE];  
 char surname[CHSIZE];  
 char birthday[CHSIZE];  
 int marks[MARKSIZE];  
 double ratingMark;  
 struct students\* next;  
} students;  
// Міняє місцями 2 елементи у списку  
void swap(students \*curr,students\*next);  
//Сортує список у порядку зростання  
void bubbleSort( students \*head);  
// виводить на екран список студентів у форматі:  
// Surname Name Birthday list of marks  
void displayList(students\* head);  
// пошук рейтингового балу кожного студента  
void findRatingMark(students\* head);  
// Вставка нового вузла у кінець списку  
void insertAtEnd(students \*head);  
//вставка нового вузла у початок списку  
void insertAtBegin(students \*\*head);  
//видалення зі списку студентів з рейтинговим балов вищим за 4.5  
void deleteUpper4\_5(students\*\*head);  
// Зчитування списку з студентів з файлу в односвязний список  
void readFromFile(students \*\*head);  
#endif

Function.c

#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#include "functions.h"  
//--------------------------------------------------------------------------------  
void swap(students \*curr,students\*next)  
{  
 char tempName[CHSIZE];  
 strcpy(tempName,curr->name);  
 strcpy(curr->name,next->name);  
 strcpy(next->name,tempName);  
  
 char tempSur[CHSIZE];  
 strcpy(tempSur,curr->surname);  
 strcpy(curr->surname,next->surname);  
 strcpy(next->surname,tempSur);  
  
 char tempBirth[CHSIZE];  
 strcpy(tempBirth,curr->birthday);  
 strcpy(curr->birthday,next->birthday);  
 strcpy(next->birthday,tempBirth);  
  
 int tempRat=curr->ratingMark;  
 curr->ratingMark=next->ratingMark;  
 next->ratingMark=tempRat;  
  
 int tempArr[5];  
 for (int i = 0; i < MARKSIZE; ++i) {  
 tempArr[i]=curr->marks[i];  
 curr->marks[i]=next->marks[i];  
 next->marks[i]=tempArr[i];  
 }  
}  
//-----------------------------------------------------------------------  
void bubbleSort( students \*head){  
 int change, i;  
 students \*current;  
 students \*before = NULL;  
  
 /\* Checking for empty list \*/  
 if (head == NULL)  
 return;  
  
 do {  
 change = 0;  
 current = head;  
  
 while (current->next != before)  
 {  
 if (current->ratingMark > current->next->ratingMark)  
 {  
 swap(current, current->next);  
 // SwapPointers(&current,&(current->next));  
 change = 1;  
 }  
 current = current->next;  
 }  
 before = current;  
 }while (change);  
 current=NULL;  
}  
//-----------------------------------------------------------------------  
void displayList(students\* head) {  
 students\* p = head;  
 while (p != NULL) {  
 printf("\t%s\t%s\t%s\t", p->name, p->surname, p->birthday);  
 for (int i = 0; i < MARKSIZE; ++i)  
 printf("%d\t", p->marks[i]);  
 p = p->next;  
 printf("\n");  
 }  
 printf("\n");  
}  
//-----------------------------------------------------------------------  
void findRatingMark(students\* head) {  
 students\* p = head;  
  
 while (p != NULL) {  
 double sum=0;  
 for (int i = 0; i < MARKSIZE; ++i) {  
 sum += (double )p->marks[i];  
 }  
  
 p->ratingMark=sum\*0.95/MARKSIZE;  
  
 printf("%s\t%s\t%s\t%lf", p->name, p->surname, p->birthday,p->ratingMark);  
 p = p->next;  
 printf("\n");  
 }  
}  
//-----------------------------------------------------------------------  
void insertAtBegin(students \*\*head){  
 int arr[5];  
 char n[20];  
 char sn[20];  
 char birth[20];  
 printf("Enter surname, name, birthday and 5 marks of inserted element:\n");  
 scanf(" %s%s%s",sn,n,birth);  
 for(int i=0;i<MARKSIZE;++i)  
 scanf("%d",&(arr[i]));  
 students \* newStudent =(students\*) malloc(sizeof(students));  
 strcpy(newStudent->name, n);  
 strcpy(newStudent->surname, sn);  
 strcpy(newStudent->birthday, birth);  
 for(int i=0;i<MARKSIZE;++i)  
 newStudent->marks[i]=arr[i];  
 newStudent->next=\*head;  
 \*head = newStudent;  
}  
//-----------------------------------------------------------------------  
void insertAtEnd(students \*head){  
 int arr[5];  
 char n[20];  
 char sn[20];  
 char birth[20];  
 scanf(" %s%s%s",sn,n,birth);  
 for(int i=0;i<MARKSIZE;++i)  
 scanf("%d",&(arr[i]));  
 students \* newStudent =(students\*) malloc(sizeof(students));  
 strcpy(newStudent->name, n);  
 strcpy(newStudent->surname, sn);  
 strcpy(newStudent->birthday, birth);  
 for(int i=0;i<MARKSIZE;++i)  
 newStudent->marks[i]=arr[i];  
  
 newStudent->next = NULL;  
 students \*currentHead=head;  
 while(currentHead->next != NULL){  
 currentHead = currentHead->next;  
 }  
 currentHead->next=newStudent->next;  
}  
//-----------------------------------------------------------------------  
void deleteUpper4\_5(students\*\*head){  
 students \*temp=\*head;  
 while(temp->next!=NULL){  
 if(temp->ratingMark<4.5)  
 temp=temp->next;  
 else{  
 for (;temp->next!=NULL;temp=temp->next){  
 students\* curr = temp->next->next;  
 free(temp->next);  
 temp->next = curr;  
 }}  
 }  
}  
//-----------------------------------------------------------------------  
void readFromFile(students \*\*head){  
 FILE\* fileInfo = NULL;  
 fileInfo = fopen("students.txt", "r");  
 students\* current = NULL;  
 students\* currentHead = NULL;  
 int arr[5];  
 char n[20];  
 char sn[20];  
 char birth[20];  
  
 while (fscanf(fileInfo, "%s%s%s", n, sn, birth) != EOF){  
 current = (students\*)malloc(sizeof(students));  
 if (current == NULL){  
 printf("Error. Memory isn't allocated");  
 return;  
 }  
 else{  
 for (int i = 0; i < MARKSIZE; ++i)  
 fscanf(fileInfo, "%d", &(arr[i]));  
 strcpy(current->surname, sn);  
 strcpy(current->name, n);  
 strcpy(current->birthday, birth);  
 for (int i = 0; i < MARKSIZE; ++i)  
 current->marks[i] = \*(arr + i);  
  
 if (\*head == NULL){  
 \*head = current;  
 (\*head)->next = NULL;  
 currentHead = \*head;  
 }  
 else {  
 current->next = NULL;  
 while (currentHead->next != NULL){  
 currentHead = currentHead->next;  
 }  
 currentHead->next = current;  
 }  
 }  
 }  
 fclose(fileInfo);  
 free(current);  
}

**Висновки**

В ході виконання даної лабораторної роботи я навчилася документувати основні результати етапів проектування та кодування найпростіших програм, а також реалізувала це на практиці.