**МIНIСТЕРСТВО  ОСВIТИ І НАУКИ  УКРАЇНИ**

**НАЦІОНАЛЬНИЙ   ТЕХНІЧНИЙ УНІВЕРСИТЕТ   УКРАЇНИ**

**“КИЇВСЬКИЙ  ПОЛІТЕХНІЧНИЙ  ІНСТИТУТ**

**ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики**

**Кафедра програмного забезпечення комп’ютерних систем**

**Лабораторна робота № 4**

**з дисципліни “ Основи програмування ”**

**тема “Основи С++. Класи”**

|  |  |  |
| --- | --- | --- |
| **Виконав**  **студент I курсу**  **групи КП-82**  **Мельничук Олексій Геннадійович**  **варіант №15** |  | **Перевірив**  **“\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.**  **викладач**  **Гадиняк Руслан Анатолійович**  **(прізвище, ім’я, по батькові)** |

**Київ 2019**

**Мета роботи**

Вивчити основні відмінності між кодом мови С та мови С++.

Навчитись створювати С++ класи для реалізації базових АТД.

**Постановка завдання**

Скопіювати у проект даного завдання рішення лабораторної роботи №1.

Змінити розширення файлів .c на .cpp (або .cc) та спробувати зібрати проект за допомогою g++ \*.cpp -std=c++17 -pedantic-errors -Wall -Werror -Wno-unused.

Виправити всі помилки компіляції, наприклад:

обов'язкове явне приведення вказівників з void \*

замінити всі VLA масиви на динамічно створені масиви

Замінити всі застарілі підключення стандартної бібліотеки С на нові (приклад: <stdlib.h> на <cstdlib>)

Замінити всі виклики malloc() на оператор new, а виклики free() на оператор delete. Вийняток: виклики всередині динамічних АТД для використання realloc().

Переписати АТД Список та АТД за варіантом з використанням С++ класів:

Змінити ключове слово struct на class (наприклад, struct \_\_List -> class \_\_List, це зробить всі поля приватними).

Клас обов'язково має бути описаний у заголовочному файлі.

Видалити рядок із typedef для типу АТД, змінити тег класу на синонім, що був у typedef (class \_\_List -> class List).

Додати до назв всіх полів класу суфікс \_ (size -> size\_).

Додати всередині класу після всіх полів розділ public:.

Прототипи публічних методів писати у цьому розділі.

Конструктори і деструктор:

Створити у класі прототипи конструктора за замовчуванням і деструктора (наприклад, List(), ~List()).

Реалізувати конструктор за замовчуванням (List::List()) і деструктор (List::~List()) перенісши до них код ініціалізацітора та деініціалізатора відповідно.

Видалити в цих функціях параметр self і замість його імені використовувати прихований вказівник this.

Видалити прототипи, реалізації і виклики ініціалізацітора, деініціалізатора, алокатора і деалокатора.

Реалізації конструкторів та деструкторів мають бути винесені у окремий файл з кодом.

Методи класу:

Внести прототипи всіх операцій у клас без префіксів типу у назві (наприклад, size\_t size();).

Змінити назву у реалізаціях цих методів з використанням оператора :: (наприклад, size\_t List::size() { ... }).

Видалити параметр self і замість його імені використовувати прихований вказівник this.

Змінити всі виклики методів з використанням оператора . (для екземплярів) або -> (для вказівників на екземпляри). Наприклад:

List\_size(&lst) -> lst.size() або

List\_size(pLst) -> pLst->size().

Реалізації всіх методів мають бути винесені у окремий файл з кодом.

Перевантаження операторів:

Замість методів get() і set() для доступу до зчитування-запису елементів колекцій реалізувати operator [] (Додаток А).

Реалізації всіх перевантажених операторів мають бути винесені у окремий файл з кодом.

У даній лабораторній роботі НЕ дозволяється використовувати:

шаблонні класи

стандартні контейнери

Опціонально можна:

Підключити <iostream> та замінити весь вивід через printf() на вивід за допомогою std::cout, std::endl та спеціального оператора вставки у потік <<.

Підключити <string> та використати тип std::string замість const char \* та char \* (якщо є використання).

**Тексти коду програм**

|  |
| --- |
| **main.cpp** |
| #include "list.hpp"  #include "stack.hpp"  #include "addon.hpp"  int main()  {  List dlist;  dlist.List\_fwrite();  dlist.List\_print();  dlist.List\_sort();  dlist.List\_print();  Stack stack1;  Stack stack2;  List dlist2;  Write\_to\_stacks(&stack1, &stack2, &dlist);  stack1.Stack\_print();  stack2.Stack\_print();  Write\_to\_list(&stack1, &dlist2);  Write\_to\_list(&stack2, &dlist2);  dlist2.List\_print();  std::cout << "End of program. " << std::endl;  return 0;  } |

|  |
| --- |
| **list.hpp** |
| #pragma once  #include <cstdio>  #include <cstdlib>  #include <cstring>  #include <iostream>  class List  {  private:  char \*\*array\_;  int capacity\_;  int length\_;  public:  List();  ~List();  void List\_add(char \*string, int len);  void List\_print();  void List\_fwrite();  void List\_sort();  void List\_realloc(int newCapacity);  int List\_length();  char\* & operator[](int index);  }; |

|  |
| --- |
| **list.cpp** |
| #include "list.hpp"  char\* & List::operator[](int index)  {  return this->array\_[index];  }  int List::List\_length()  {  return this->length\_;  }  void List::List\_realloc(int newCapacity)  {  char \*\*newArray = (char\*\*)realloc((this->array\_), sizeof(char\*) \* newCapacity);  if (newArray == NULL)  {  free(this->array\_);  std::cerr << "Reallocation error\n" << std::endl;  abort();  }  this->array\_ = newArray;  this->capacity\_ = newCapacity;  }  void List::List\_add(char \*string, int len)  {  this->array\_[this->length\_] = (char\*)malloc((len + 1) \* sizeof(char));  strcpy(this->array\_[this->length\_], string);  //  this->length\_ += 1;  //  if (this->length\_ == this->capacity\_)  {  int newCapacity = this->capacity\_ + 1;  this->List\_realloc(newCapacity);  }  }  void List::List\_print()  {  std::cout << "Printing list:\n " << std::endl;  for(int i = 0; i < this->length\_; i++)  {  std::cout << this->array\_[i] << std::endl;  }  puts("");  }  void List::List\_sort()  {  std::cout << "Sorting...\n " << std::endl;  char \*bufpoint;  for(int i = 0; i < this->length\_; i++)  {  if((strstr(this->array\_[i], " the ") != NULL))  {  bufpoint = this->array\_[i];  for(int j = i; j < this->length\_; j++)  {  if((strstr(this->array\_[j], " the ") == NULL))  {  this->array\_[i] = this->array\_[j];  this->array\_[j] = bufpoint;  break;  }  }  }  }  }  void List::List\_fwrite()  {  FILE \*file;  char buf[256] = {'\0'};  int bufc = 0;  file = fopen("data.txt", "r");  while (fgets(buf, 256, file))  {  while (buf[bufc] != '\0')  {  if (buf[bufc] == '\n') buf[bufc] = '\0';  bufc++;  }  this->List\_add(buf, bufc);  for(int i = 0; i < bufc; i++)  {  buf[i] = '\0';  }  bufc = 0;  }  fclose(file);  } |

|  |
| --- |
| **stack.hpp** |
| #pragma once  #include <cstdio>  #include <cstdlib>  #include <cstring>  #include <iostream>  class Stack  {  private:  char \*\*array\_;  int capacity\_;  int length\_;  public:  Stack();  ~Stack();  void Stack\_push(char \*string, int len);  char \*Stack\_pop();  char \*Stack\_peek();  bool Stack\_isEmpty();  void Stack\_print();  void Stack\_realloc(int newCapacity);  int Stack\_length();  char\* & operator[](int index);  }; |

|  |
| --- |
| **stack.cpp** |
| #include "stack.hpp"  char\* & Stack::operator[](int index)  {  return this->array\_[index];  }  int Stack::Stack\_length()  {  return this->length\_;  }  void Stack::Stack\_realloc(int newCapacity)  {  char \*\*newArray = (char\*\*)realloc(this->array\_, sizeof(char\*) \* newCapacity);  if (newArray == NULL)  {  for (int i = 0; i < this->length\_; i++)  {  free(this->array\_[i]);  }  free(this->array\_);  std::cerr << "Reallocation error\n" << std::endl;  abort();  }  this->array\_ = newArray;  this->capacity\_ = newCapacity;  }  void Stack::Stack\_push(char \*string, int len)  {  this->array\_[this->length\_] = (char\*)malloc((len + 1) \* sizeof(char));  strcpy(this->array\_[this->length\_], string);  this->length\_ += 1;  //  if (this->length\_ == this->capacity\_)  {  int newCapacity = this->capacity\_ + 1;  this->Stack\_realloc(newCapacity);  }  }  char \* Stack::Stack\_pop()  {  if (this->length\_ == 0)  {  std::cerr << "Stack underflow\n" << std::endl;  abort();  }  char \*string = this->array\_[this->length\_ - 1];  this->length\_ -= 1;  return string;  }  char \* Stack::Stack\_peek()  {  if (this->length\_ == 0)  {  std::cerr << "Stack underflow\n" << std::endl;  abort();  }  char \*string = this->array\_[this->length\_ - 1];  return string;  }  bool Stack::Stack\_isEmpty()  {  return this->length\_ == 0;  }  void Stack::Stack\_print()  {  std::cout << "Printing stack: \n" << std::endl;  for(int i = 0; i < this->length\_; i++)  {  std::cout << this->array\_[i] << std::endl;  }  puts("");  } |

|  |
| --- |
| **addon.hpp** |
| #pragma once  #include "stack.hpp"  #include "list.hpp"  void Write\_to\_stacks(Stack \*self1, Stack \*self2, List \*self);  void Write\_to\_list(Stack \*stacks, List \*dlist); |

|  |
| --- |
| **addon.cpp** |
| #include "addon.hpp"  void Write\_to\_stacks(Stack \*self1, Stack \*self2, List \*self)  {  std::cout << "Writing into stacks...\n " << std::endl;  for(int i = 0; i < self->List\_length(); i++)  {  if(i % 2 == 0)  {  self1->Stack\_push(self->operator[](i), (int)strlen(self->operator[](i)));  }  else if(i % 2 != 0)  {  self2->Stack\_push(self->operator[](i), (int)strlen(self->operator[](i)));  }  }  }  void Write\_to\_list(Stack \*stack, List \*dlist)  {  std::cout << "Writing into list...\n " << std::endl;  for(int i = 0; i < stack->Stack\_length(); i++)  {  dlist->List\_add(stack->operator[](i), (int)strlen(stack->operator[](i)));  }  }  List::List()  {  this->length\_ = 0;  this->capacity\_ = 2;  this->array\_ = (char \*\*)malloc(this->capacity\_ \* sizeof(char \*));  if (this->array\_ == NULL)  {  std::cerr << "Allocation error\n" << std::endl;  abort();  }  }  List::~List()  {  for(int i = 0; i < this->length\_; i++)  {  free(this->array\_[i]);  }  free(this->array\_);  }  Stack::Stack()  {  this->length\_ = 0;  this->capacity\_ = 4;  this->array\_ = (char \*\*)malloc(this->capacity\_ \* sizeof(char \*));  if (this->array\_ == NULL)  {  std::cerr << "Allocation error\n" << std::endl;  abort();  }  }  Stack::~Stack()  {  for (int i = 0; i < this->length\_; i++)  {  free(this->array\_[i]);  }  free(this->array\_);  } |

|  |
| --- |
| **data.txt** |
| As I walk through the valley of the shadow of death  I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  Been spendin' most their lives  Livin' in the gangsta's paradise  Been spendin' most their lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise |

**Робота коду**

Створюються усі потрібні екземпляри класів.

Реалізується виділення пам’яті для масиву вказівників на строки.

Відкривається файловий потік.

Текстовий файл зчитується та додається у буфер, знаходиться розмір строки, виділяється пам’ять для одної строки списку, вміст буфера передається у строку та очищується.

Файловий потік закривається.

Список сортується відповідно до умови шляхом переставляння вказівників.

Створюються два стеки, виділяється пам’ять.

Вміст списку передається у стеки відповідно до умови, виділяється пам’ять до кожної строки у стеках.

Створюється новий список, виділяється пам’ять.

Вміст кожного стека по черзі наповнює новий список, виділяється пам’ять.

Деструктори знищують елементи класів.

Код розбивається на інтерфейс та реалізації у окремі header-файли і зв’язується.

**Приклади результатів**

Результат у консолі:

|  |
| --- |
| aleksey@auv:~/Desktop/lab4$ g++ -g \*.cpp -std=c++17 -pedantic-errors -Wall -Werror -Wno-unused  aleksey@auv:~/Desktop/lab4$ valgrind --leak-check=full ./a.out  ==9400== Memcheck, a memory error detector  ==9400== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.  ==9400== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info  ==9400== Command: ./a.out  ==9400==  Printing list:    As I walk through the valley of the shadow of death  I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  Been spendin' most their lives  Livin' in the gangsta's paradise  Been spendin' most their lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Keep spendin' most our lives  Livin' in the gangsta's paradise  Tell me why are we so blind to see  That the ones we hurt are you and me?  Sorting...    Printing list:    I take a look at my life and realize there's nothin' left  ‘Cause I've been blastin' and laughin' so long  That even my momma thinks that my mind is gone  But I ain't never crossed a man that didn't deserve it  Me be treated like a punk, you know that's unheard of  You better watch how you talkin' and where you walkin'  Or you and your homies might be lined in chalk  I really hate to trip, but I gotta loc  Been spendin' most their lives  Been spendin' most their lives  Keep spendin' most our lives  Keep spendin' most our lives  Tell me why are we so blind to see  Livin' in the gangsta's paradise  As they croak, I see myself in the pistol smoke  Livin' in the gangsta's paradise  Fool, I'm the kinda G the little homies wanna be like  Livin' in the gangsta's paradise  On my knees in the night, sayin' prayers in the streetlight  Livin' in the gangsta's paradise  As I walk through the valley of the shadow of death  That the ones we hurt are you and me?  Writing into stacks...    Printing stack:  I take a look at my life and realize there's nothin' left  That even my momma thinks that my mind is gone  Me be treated like a punk, you know that's unheard of  Or you and your homies might be lined in chalk  Been spendin' most their lives  Keep spendin' most our lives  Tell me why are we so blind to see  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  As I walk through the valley of the shadow of death  Printing stack:  ‘Cause I've been blastin' and laughin' so long  But I ain't never crossed a man that didn't deserve it  You better watch how you talkin' and where you walkin'  I really hate to trip, but I gotta loc  Been spendin' most their lives  Keep spendin' most our lives  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  That the ones we hurt are you and me?  Writing into list...    Writing into list...    Printing list:    I take a look at my life and realize there's nothin' left  That even my momma thinks that my mind is gone  Me be treated like a punk, you know that's unheard of  Or you and your homies might be lined in chalk  Been spendin' most their lives  Keep spendin' most our lives  Tell me why are we so blind to see  As they croak, I see myself in the pistol smoke  Fool, I'm the kinda G the little homies wanna be like  On my knees in the night, sayin' prayers in the streetlight  As I walk through the valley of the shadow of death  ‘Cause I've been blastin' and laughin' so long  But I ain't never crossed a man that didn't deserve it  You better watch how you talkin' and where you walkin'  I really hate to trip, but I gotta loc  Been spendin' most their lives  Keep spendin' most our lives  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  Livin' in the gangsta's paradise  That the ones we hurt are you and me?  End of program.  ==9400==  ==9400== HEAP SUMMARY:  ==9400== in use at exit: 0 bytes in 0 blocks  ==9400== total heap usage: 132 allocs, 132 frees, 86,778 bytes allocated  ==9400==  ==9400== All heap blocks were freed -- no leaks are possible  ==9400==  ==9400== For counts of detected and suppressed errors, rerun with: -v  ==9400== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)  aleksey@auv:~/Desktop/lab4$ |

**Висновки**

Дана лабораторна робота допомогла опрацювати базові знання мови C++, зрозуміти основні відмінності мови від C.

Були опрацьовані класи, їх створення та знищення, концепт публічності та приватності методів та властивостей.

Також було опрацьоване явне приведення, вивід за допомогою потокових об’єктів та оператора <<, перевантаження оператора індексації.