**Міністерство освіти і науки України**

**Національний університет «Запорізька політехніка»**

кафедра програмних засобів

**//ЗВІТ**

з лабораторної роботи №2

з дисципліни «Об'єктно-орієнтоване програмування»

на тему : «ДИНАМІЧНІ КЛАСОВІ ТИПИ»

Варіант №7

Виконала:

студент групи КНТ – 132 Сімак А.В.

Перевірив:

Старший викладач Качан О. І.

2023

**1.1 Мета роботи:**

Навчитись використовувати динамічні класи при створенні програм.

**1.2 Завдання до роботи.**

Варіант 7. Створити динамічний клас для роботи з рядком - масивом символів. Максимальна довжина рядка – 65535 символів. Передбачити функції для виконання наступних операцій: ініціалізація ASCIIZ–рядка (рядка, що завершується нуль–байтом); введення з клавіатури; виведення на екран; повернення кількості символів у рядку; конкатенація рядків; пошук підрядка; пошук і заміна підрядка.

**1.3 Текст програми.**

#include <iostream>

#include <cstring>

using namespace std;

class String {

char\* s; // Рядок, який буде зберігатися в об'єкті

public:

// Конструктор класу

String(const char\* str="") {

s = new char[strlen(str)+1];// Виділення пам'яті для рядка та копіювання переданого рядка

strcpy(s, str);

}

// Деструктор класу

~String() {

delete[] s;

}

// Ініціалізація з ASCIZ–рядка

void asciz(const char\* str) {// Видалення останнього символу - нуль-байту

int len = strlen(str);

s = new char[len];

strncpy(s, str, len-1); //копіювання

s[len-1] = '\0'; //нульбайт на останнє місце в рядку

}

// Введення з клавіатури

void input() {// Запит рядка від користувача

char arr[65536];

cout<<"vvedit vash ryadok : "<<endl;

cin.getline(arr, 65536);

s = new char[strlen(arr)+1]; // створ. нової розміром старої + 0 байт

strcpy(s, arr); //копіювання одного віньше

}

// Виведення на екран

void output() {// Вивід рядка на екран

cout <<"Vash ryadok : "<< s << endl;

}

// Повернення кількості символів у рядку

int length() {

cout<<"Dovgina ryadka : ";

return strlen(s);

}

// Конкатенація рядків

void concatenate(String other) {

char\* newstr = new char[strlen(s) + strlen(other.s) + 1];

strcpy(newstr, s); //копіюєм 1 в іньше

strcat(newstr, other.s); // об'єднання

delete[] s;

s = newstr; //переприсвоєння

}

// Пошук підрядка

int find(String substr) {// Пошук першого входження підрядка в рядку

char\* ptr = strstr(s, substr.s); // пошук покажчик на 1 входження

if (ptr == nullptr) {

return -1; // якщо немає то повертає -1

}

return ptr - s; // якщо є то повертає різницю покажчиків

}

// Пошук і заміна підрядка

void replace(String oldsub, String newsub) { // Заміна всіх входжень підрядка на новий підрядок

char\* ptr = strstr(s, oldsub.s); //пошук покажчика га 1 входження

while (ptr != nullptr) {

char\* newstr = new char[strlen(s) + strlen(newsub.s) - strlen(oldsub.s) + 1]; //нова строка довжиною ісходної + довжина нового підрядка і - того який був

strncpy(newstr, s, ptr - s); // копіюємо в новий рядок частину вихідного рядка

newstr[ptr - s] = '\0'; //додаємо нуль символ

strcat(newstr, newsub.s); // додаємо в новий рядок новий підрядок

strcat(newstr, ptr + strlen(oldsub.s)); //додаємо в новий рядок залишок рядка, наступний за знайденим входженням старого підрядка

delete[] s; //видаляємо стару строку

s = newstr; // присвоюємо покажчик на новий рядок s, тому що ми замінюємо старий підрядок на новий

ptr = strstr(s, oldsub.s); // шукаємо наступне входження старого підрядка в новому рядку

}

}

};

int main() {

String s1;

cout<<"Ryadok No 1 ";

s1.input();

String s2;

cout<<"Dlya konkatenacii ";

s2.input();

String s3;

s1.output(); // Виведе "Vash ryadok : рядок

s2.output(); // Виведе "Vash ryadok : рядок

s3.concatenate(s1); // Додасть до порожнього рядка s3 рядок s1

s3.concatenate(s2); // Додасть до s3 рядок s2

cout<<"Pislya konkatenatciyi ";

s3.output(); // Виведе "Vash ryadok : Hello, world!"

String s4;

cout<<"Dlya vuznachennya rozmiry ";

s4.input(); // Запит рядка від користувача

s4.output(); // Виведе введений рядок

int len = s4.length(); // Збереже довжину введеного рядка у змінну len

cout << "Dovgina ryadka : " << len << endl;

String s5;

cout<<"Dlya vyznachennya pozytsiyi ";

s5.input();

s5.output();

String s6;

cout<<"Pidryadoc ";

s6.input();

s6.output();

int index = s5.find(s6); // Збереже позицію першого входження підрядка у змінну index

cout << "Pozitsiya pidryadka: " << index << endl;

String s7;

cout<<"Dlya zminy fragmenty v ryadky ";

s7.input();

s7.output();

String s8;

cout<<"Pidryadoc yakyy zminyty ";

s8.input();

s8.output();

String s9;

cout<<"Pidryadoc na yakyy zaminyty ";

s9.input();

s9.output();

s7.replace(s8, s9); // Замінить у рядку всі входження

s7.output(); // Виведе "Vash ryadok : і змінений

return 0;

}

**1.4 Результати роботи програми.**

Скріншоти роботи програми:

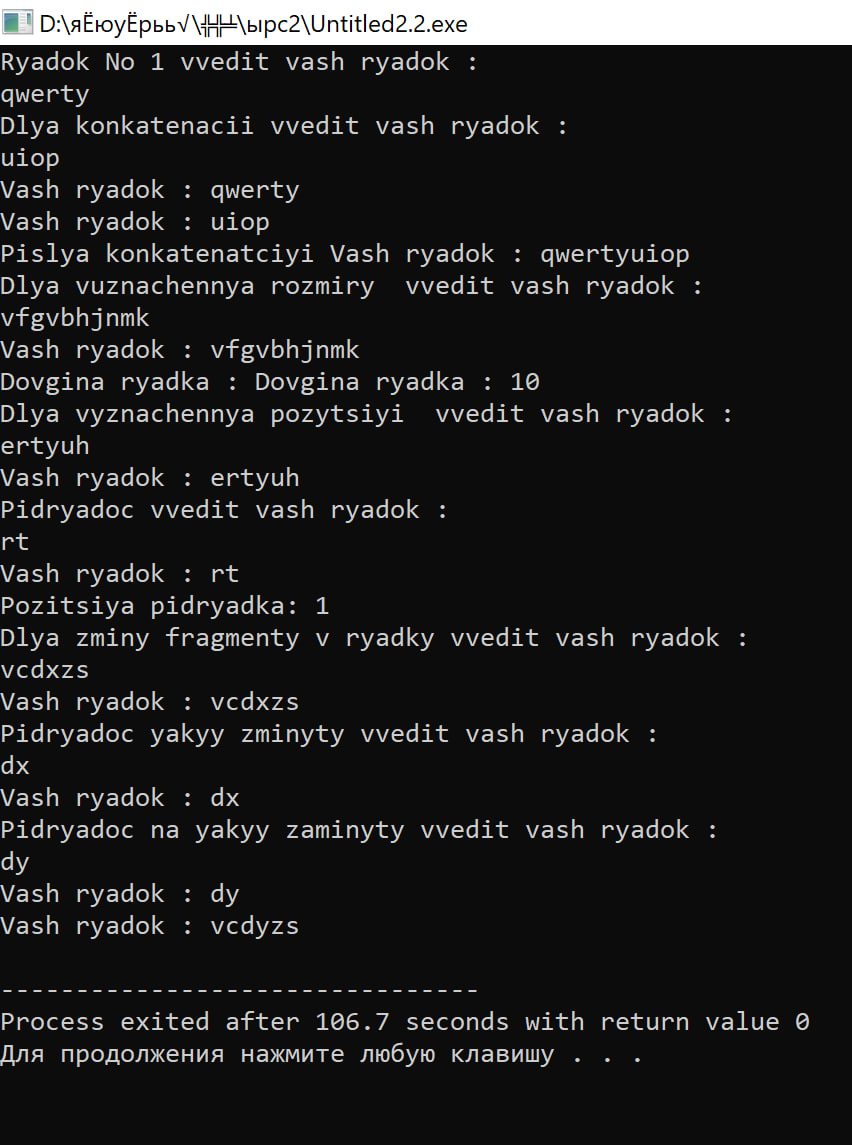


Рисунок 1.4 - Скріншоти роботи програми:

**1.5 Висновок:**

Під час виконання лабораторної роботи було отримано досвід використання динамічних класів для створення програм. Було з'ясовано, що використання динамічних класів дозволяє динамічно змінювати розмір об'єктів під час виконання програми, що є дуже зручним. Крім того, було досліджено різні способи створення динамічних класів та їх використання, зокрема вказівники, оператор new та delete, конструктор копіювання тощо.

Отже, використання динамічних класів є важливим елементом при створенні програм, які вимагають динамічного змінення розміру об'єктів під час виконання. Динамічні класи також дозволяють зменшити витрати пам'яті та збільшити ефективність програми.

**1.6 Відповіді на контрольні питання:**

**1. Чим відрізняється динамічний клас від статичного класу??**

*Статичний клас визначається на етапі компіляції і зберігається в пам'яті, що призначене для статичних об'єктів. Його розмір відомий заздалегідь, і він не може змінюватися під час виконання програми. Об'єкти статичного класу створюються на стеку або в області пам'яті, що призначена для статичних змінних.*

*Динамічний клас, з іншого боку, визначається виключно під час виконання програми і зберігається в області пам'яті, що призначена для динамічно виділених об'єктів. Розмір динамічного класу може змінюватися під час виконання програми, і об'єкти динамічного класу створюються в області пам'яті, що призначена для динамічно виділених об'єктів, таких як купа.*

*Таким чином, головною відмінністю між статичним і динамічним класом є те, що статичний клас визначається на етапі компіляції і його розмір не може змінюватися під час виконання програми, тоді як динамічний клас визначається виключно під час виконання програми і може змінюватися під час виконання програми.*

**11. Чи можно об'єкти класу передавати за посиланням?**

*З допомогою посилання можна передавати змінну чи об'єкт класу у функцію. Таким чином, у тілі функції можна змінювати значення змінної, що була передана.*

**10.Який механізм повернення значень типу клас***?*

*Повернення значень типу клас відбувається за допомогою оператора return, який може повертати значення типу клас через об'єкт класу, або через вказівник на об'єкт класу. Якщо тип повертається через об'єкт класу, то цей об'єкт буде повернуто за значенням, якщо через вказівник на об'єкт класу - то буде повернуто адресу пам'яті, на яку вказує вказівник.*

Person getPerson() {

Person p("John", 25);

return p; // Повертаємо об'єкт типу Person

}

Якщо хочемо повернути об'єкт цього класу з функції

*Якщо ж хочемо повернути адресу пам'яті на об'єкт класу, то можна зробити наступним чином:*

Person\* getPerson() {

Person\* p = new Person("John", 25);

return p; // Повертаємо вказівник на об'єкт типу Person

}

**7. На що вказує покажчик \*this?**

*Покажчик \*this вказує на поточний об'єкт класу, в якому він використовується. Це дає можливість звернутися до полів та методів класу з середини його ж методів, що може бути корисно в деяких ситуаціях. Наприклад, можна використовувати \*this для повернення значення об'єкта з методу класу, або для передачі об'єкта як параметра в інші методи класу.*

***5. Коли використовується конструктор копіювання?***

*Конструктор копіювання використовується для створення нового об'єкту, який є копією існуючого об'єкту. В С++, конструктор копіювання автоматично генерується компілятором, якщо він не визначений явно в класі. Цей конструктор викликається, коли об'єкт передається за значенням, коли створюється копія об'єкту, а також при використанні деяких стандартних функцій, таких як std::vector::push\_back() або std::make\_shared().*