Міністерство освіти і науки

Національний університет “Львівська політехніка”

**Кафедра ЕОМ**



**Звіт**

з лабораторної роботи № 1

# з дисципліни: “ Програмування, частина 2 (Об'єктно-орієнтоване програмування”

на тему: “ ПЕРЕВАНТАЖЕННЯ ФУНКЦІЙ ”

Виконав: ст. гр. КІ-15 Гвоздь Ю. А.

Прийняв: Козак Н. Б.

Львів – 2020

Мета: познайомитися із перевантаженням функцій.

ТЕОРЕТИЧНІ ВІДОМОСТІ

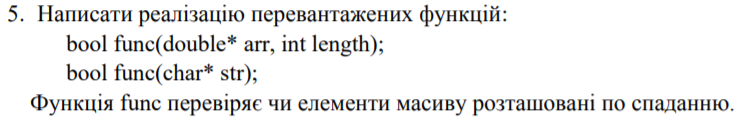
Перевантаження функцій

Перевантаження функцій У мові С++, на відміну від мови С дозволяється визначати декілька функцій з одним і тим же іменем за умови, що дані функції мають різну сигнатуру (різні типи та кількість аргументів функції). При цьому тип значення що повертається при перевантаженні до уваги не приймається. При виклику функції з ім‘ям func спершу компілятор намагатиметься знайти функцію, формальні аргументи якої відповідають фактичним аргументам без усяких перетворень типів або з використанням тільки неминучих перетворень - наприклад, імені масиву до покажчика або значення змінної до константи або навпаки. Якщо відповідна функція не знайдена, то здійснюється пошук такої функції, щоб для відповідності формальних і фактичних аргументів досить було використати тільки такі стандартні перетворення, що не спричиняють перетворень цілих типів до типів з плаваючою крапкою і навпаки. При цьому підбирається функція, для якої число таких перетворень було б мінімальним. Третім етапом є пошук такої функції, для виклику якої досить здійснити будь-які стандартні перетворення аргументів (і знову так, щоб цих перетворень було якнайменше). Далі здійснюється пошук функції, для якої аргументи можна одержати за допомогою всіх перетворень, розглянутих до цього, а також перетворень типів, визначених самим програмістом. Якщо й у цьому випадку єдина потрібна функція не знайдена, то на останньому етапі компілятор пробує знайти відповідність з урахуванням списку невизначених аргументів. Так, виклик функції func (1, 2, 3) може бути співставлений лише з функцією, що оголошена як іnt func(float, ...). Якщо компілятор не знайде жодної підходящої функції, або виклик функції не може бути однозначно співставлений з однією з оголошених функцій, то програма не скомпілюється і буде виведене повідомлення про помилку. Зверніть увагу, що співставлення викликів функцій з оголошеними в програмі функціями відбувається на етапі компіляції, а не в процесі виконання програми.

Вбудовані (іnlіne) функції

У мові С директива препроцесора #defіne використовується для визначення констант та макросів (макровизначень). Наприклад, директива #defіne sqr(x) ((x)\*(x)) оголошує макрос, що дозволяє обчислювати квадрат від фактичного аргументу, який підставляється замість формального аргументу х. У даному випадку скрізь у тексті програми під час компіляції замість sqr(і) буде підставлено вираз ((і)\*(і)). Слід зазначити, що аргумент при використанні макровизначення може бути будь-яким (змінною будь-якого значущого типу, константою, функцією, що повертає значення, іншим макросом). Це дуже схоже на виклик функції, але насправді відбувається лише текстова заміна. З цієї причини макроси працюють швидше функцій, оскільки при їх використанні не затрачаються ресурси на виклик функції. Але по цій же причині вони є потенційним джерелом помилок. У випадку sqr(і++) змінна і інкрементуватиметься не один, а два рази. Мова С++ пропонує безпечну заміну макросам – вбудовані (іnlіne) функції. На відміну від макросів, вбудовані функції не піддаються помилкам подвійного обчислення. Крім того типи аргументів перевіряються компілятором і при потребі виконуються всі необхідні перетворення. Таким чином, якщо у вас є маленька функція (кілька рядків коду), яку доводиться часто викликати, то її можна оголосити як іnlіne. Оголошена в такий спосіб функція буде проаналізована компілятором на можливість її реалізації у вигляді вбудованої функції. Якщо компілятор вважатиме доцільним реалізувати її як вбудовану, то дана функція не буде викликатися. Замість цього тіло функції підставлятиметься в те місце програми, де здійснюється виклик. При цьому підвищується ефективність програми ціною збільшення розміру коду програми. Якщо ж компілятор вважатиме, що дана функція завелика для її ефективної реалізації як вбудованої, то вона буде реалізована як звичайна функція. Таким чином кінцеве рішення у питанні робити функцію позначену як inline вбудованою чи ні належить компілятору. Щоб оголосити функцію як іnlіne необхідно просто поставити ключове слово іnlіne перед оголошенням функцї.

**Завдання:**



**Виконання завдання:**

Код програми:

#include <iostream>

#include <iomanip>

#include <cstdlib>

using namespace std;

bool func(double\* arr, int length) {

for (int c = 0; c < length - 1; ++c) {

if (arr[c] <= arr[c + 1]) {

return false;

}

}

return true;

}

bool func(char\* str) {

for (int c = 0; c < strlen(str) - 1; ++c) {

if (str[c] <= str[c + 1]) {

return false;

}

}

return true;

}

int main() {

double d1[] = { 4, 3, 2, 1 };

double d2[] = { 4, 2, 3, 1 };

char c1[] = "dcba";

char c2[] = "dbca";

cout << boolalpha;

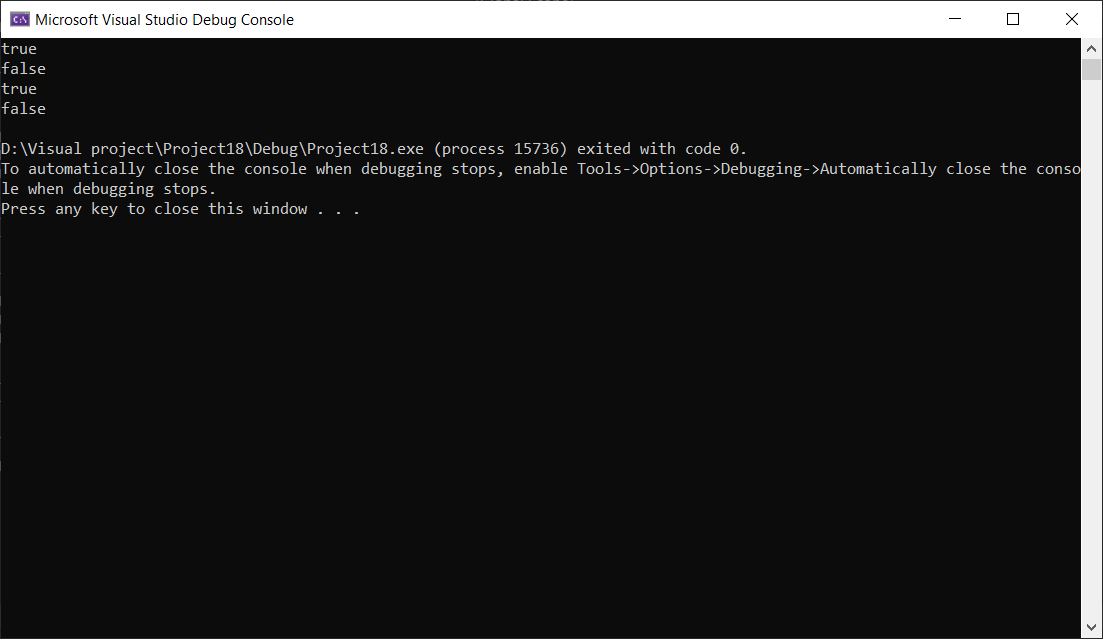
cout << func(d1, 4) << endl;

cout << func(d2, 4) << endl;

cout << func(c1) << endl;

cout << func(c2) << endl;

}

Результат роботи програми: 

Висновок: на цій лабораторній роботі я познайомився із перевантаженням функцій.