Міністерство освіти і науки України

Національний лісотехнічний університет України

Кафедра інженерії програмного забезпечення

**ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №7**

з навчальної дисципліни

«Об’єктно – орієнтоване програмування»

на тему: «Стандартна бібліотека шаблонів STL, контейнери та ітератори»

Виконав:

студент групи ІПЗС-11

Возьний С.П.

Перевірила:

Головата С. Б.

Львів – 2023

**Лабораторна робота №7**

**Тема:** Стандартна бібліотека шаблонів STL, контейнери та ітератори

**Мета:** засвоїти технологію узагальненого об’єктного програмування з використанням бібліотеки стандартних шаблонів (STL) мови C++ для роботи з вбудованими і користувацькими типами даних.

**Хід виконання**

**Завдання 1.** Створити клас Employee про облік робочого часу працівників: кількість відпрацьованих годин, погодинна оплата. Визначити метод для визначення величини зарплати працівника. Згенерувати випадкові значення відпрацьованого часу та значення погодинної оплати. Зберегти ці значення у контейнері vector <Employee\*>. Обчислити сумарні витрати компанії на оплату праці.

**Лістинг програми:**

//Файл Employee.h

#pragma once

#include <string>

using namespace std;

class Employee{

string name;

int countTimeWork;

int hourseSalary;

public:

Employee();

Employee(string name);

int sizePriceEmployee();

friend ostream& operator <<(ostream& ios, Employee& ob);

};

//Файл Employee.cpp

#include "Employee.h"

#include <cstdlib>

#include <iostream>

using namespace std;

Employee::Employee() {

name = "";

countTimeWork = 0;

hourseSalary = 0;

}

Employee::Employee(string name) {

this->name = name;

countTimeWork =rand()%101;

hourseSalary = 100+rand() % 701;

}

int Employee::sizePriceEmployee() {

return countTimeWork \* hourseSalary;

}

ostream& operator<<(ostream& os, Employee& ob) {

os << "Ім'я працівника: " << ob.name << endl;

os << "Відпрацбовані години: " << ob.countTimeWork << endl;

os << "Погодинна ставка: " << ob.hourseSalary << endl;

os << "Зарплата: "<< ob.sizePriceEmployee()<<endl;

return os;

}

//Файл main.cpp

#include <iostream>

#include "Windows.h"

#include <vector>

#include "Employee.h"

using namespace std;

int SummSalary(vector<Employee> vec) {

int tmpSumm = 0;

for (vector<Employee>::iterator i = vec.begin(); i < vec.end(); i++) {

tmpSumm += i->sizePriceEmployee();

}

return tmpSumm;

}

int main(){

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

srand(static\_cast<unsigned>(std::time(0)));

Employee person[5] = { Employee("Сергій"),Employee("Микола") ,Employee("Андрій"),Employee("Семен"),Employee("Петро") };

vector<Employee> vec;

for (int i = 0; i < 5; i++) {

vec.push\_back(person[i]);

}

cout << "Заповнений вектор:" << endl;

for (int i = 0; i < vec.size(); i++) {

cout << "Працівник №"<<i+1 << endl;

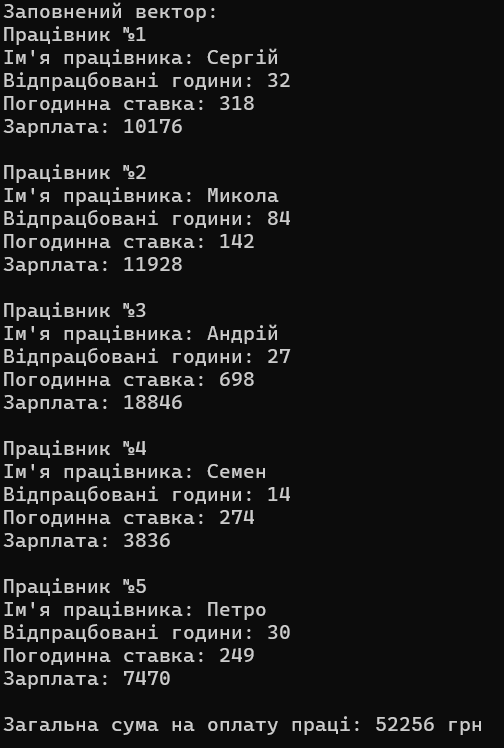
cout << vec[i] << endl;;

}

cout << "Загальна сума на оплату праці: "<<SummSalary(vec)<<" грн" << endl;

}

**Результати виконання програми:**

**Контрольні запитання**

1. **Які засоби містить стандартна бібліотека шаблонів?**

Дана бібліотека містить контейнери, ітератори, алгоритми, алокатори, функти, пердикати, адаптери.

1. **Чи всі контейнери працюють з ітератором довільного доступу?**

З цим ітераторм працюють лише вектор та черга з двома кінцями.

1. **Для чого використовується контейнер STL?**

контейнер – об’єкт, який містить інші об’єкти, організовані у

вигляді послідовностей (колекції об’єктів);

1. **Який зв'язок між контейнерами, літераторами і алгоритмами?**

Контейнери — це структури даних, які зберігають елементи. Ітератори дозволяють доступ до елементів контейнерів. Вони працюють як вказівники. Алгоритми в STL — це функції, які виконують операції над контейнерами з використанням ітераторів.

1. **Що таке предикат і що він повертає?**

предикат – функція булевого типу, яка перевіряє визначені програмістом властивості або відношення між об’єктами;

1. **Що потрібно додати до програми, щоб створити контейнер користувацького типу і працювати з ним?**

Щоб це зробити потрібно визначити клас користувацького типу, на його основі створити контейнер заповнити його даними користувацького типу і визначити певні методи для роботи з цим контейнером (перзавантаження операторів тощо).

1. **Поясніть як працює контейнер з двоспрямованим послідовним доступом? Наведіть приклад.**

Контейнер з двонаправленим послідовним доступом, як list, дозволяє ітерувати через його елементи в обидва напрямки: вперед і назад. Цей тип контейнера ідеально підходить для сценаріїв, коли вам потрібно часто вставляти та видаляти елементи всередині списку.

1. **Що таке асоціативний контейнер?**

Асоціативні контейнери (associative containers) дозволяють за допомогою ключів отримати швидкий доступ до значень, що зберігаються в

них.