 МІНІСТЕРСТВО ОСВІТИ I НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ   
ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ

КАФЕДРА БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**Лабораторна робота №2**

з дисципліни «Алгоритмізація та програмування

на тему: «Перелічуваний тип даних»

Варіант №

**Виконав:**

**Перевірив:**

Зараховано від \_\_\_.\_\_\_.\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис викладача)

Київ-2022

**Завдання**

Відповідно до свого варіанту розробити програмний застосунок, у якому створюється змінна перелічуваного типу даних, їй присвоюється значення, введене з клавіатури, і далі це значення виводиться на екран (визначення перелічуваного типу даних виконати всередині власноруч створеного простору імен, а роботу зі змінною перелічуваного типу даних – поза межами цього простору імен функції main; під час введення значень змінної має використовуватись символьна інформація, а не значення нумератора перерахування): 

**Код програми**

#include <iostream>

namespace oceansNS

{

enum oceans { INVALID, ARCTIC, ATLANTIC, INDIAN, PACIFIC, SOUTHERN };

}

using namespace std;

int main()

{

int input\_num = 0;

const char\* output\_str = "";

oceansNS::oceans ocean = oceansNS::INVALID;

cout << "Enter a number: ";

cin >> input\_num;

ocean = static\_cast<oceansNS::oceans>(input\_num);

switch (ocean)

{

case oceansNS::ARCTIC:

output\_str = "Arctic";

break;

case oceansNS::ATLANTIC:

output\_str = "Atlantic";

break;

case oceansNS::INDIAN:

output\_str = "Indian";

break;

case oceansNS::PACIFIC:

output\_str = "Pacific";

break;

case oceansNS::SOUTHERN:

output\_str = "Southern";

break;

default:

output\_str = "Invalid";

break;

}

if (output\_str == "Invalid")

cout << "error" << endl;

else

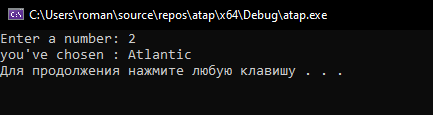
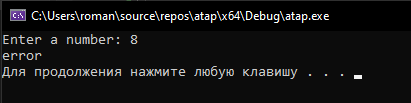
cout << "you've chosen : " << output\_str << endl;

system("pause");

return 0;

}

**Результат виконання програми:**

****

**Контрольні питання**

1. Нумератор перерахування може приймати цілочисельні значення, тобто типи даних bool, short , int , long, також до них відносять і char.
2. Це можливо лише за умови ручного введення енумераторів.
3. Два нумератори з одним ім’ям в одному перер ахування бути не можуть.
4. Конфлікт імен виникає, коли два ідентифікатори знаходяться в одній області видимості, і компілятор не може зрозуміти, який з них двох слід використовувати в конкретній ситуації. Компілятор або лінкер видасть вам помилку, оскільки у нього недостатньо інформації, щоб вирішити цю неоднозначність. Як тільки програми збільшуються в об’ємах, збільшується кількість ідентифікаторів, отже, ймовірність виникнення конфліктів імен збільшується також. Можуть виникати конфлікти імен через те, що імена нумераторів повторюються, навіть якщо вони в різних перерахуваннях.
5. Щоб знати яка кількість нумераторів зберігається у конкретному перерахуванні треба додавати до імені нумератора число, або можна присвоїти значення 1 першому нумератору, тоді значення останнього нумератора буде дорівнювати кількості нумераторів в перерахуванні.
6. Особливість ідентіфікатору з глобального контексту програми є те, що він не належить жоднії області видимості. Ідентифікатор, оголошений будь-якого визначення поза функцій, процедур, типу, є глобальним. Тобто ідентіфікатор доступний в усьому тексті програми.
7. Додання в область видимості одного ідентифікатора з простора імен;

Використання повного імені ідентифікатора зі звертанням до конкретного простору імен;

Додання всіх ідентифікаторів певного простору імен в область видимості.

1. Простори імен можуть бути вкладеними. Це означає, що один простір імен може включати в себе інший простір імен.

#include <iostream>

namespace Boo

{

namespace Doo

{

const int g\_x = 7;

}

}

int main()

{

std::cout << Boo::Doo::g\_x;

return 0;

}

1. За умови, що вкладений простір імен був оголошений перед тим, як члени зовнішнього простору імен звертаються до членів вкладеного простору імен.
2. Змінні, які оголошені в просторі імен створюються в точці визначення і знищуються при виході з блоку, в якому вони визначені. В той час, коли глобальні змінні створюються при запуску програми знищуються при її завершенні.