КІТ-119а

Автор: Чугунов В.

**ЛАБОРАТОРНА РОБОТА №4. ВНУТРІШНЄ ПОДАННЯ ІНТЕГРОВАНИХ СТРУКТУР ДАНИХ**

**Мета:** отримати та закріпити знання про внутрішнє подання інтегрованих структур даних у мовах програмування.

**Індивідуальне завдання**

Написати програму, яка виводить на екран внутрішнє подання структури з варіантною частиною та з бітовими полями, а також масива структур. Перелік властивостей та відповідні типи полів для об’єктів (рис. 4.1) .

Дослідити, як виконується вирівнювання даних та полів структури.

Порівняти час доступу до даних з вирівнюванням та за умови відсутності вирівнювання. По результатах роботи підготувати звіт з лабораторної роботи, де навести отримані результати та дати по ним пояснення, зробити висновки.

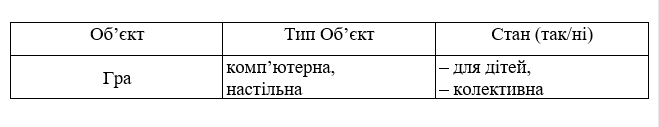
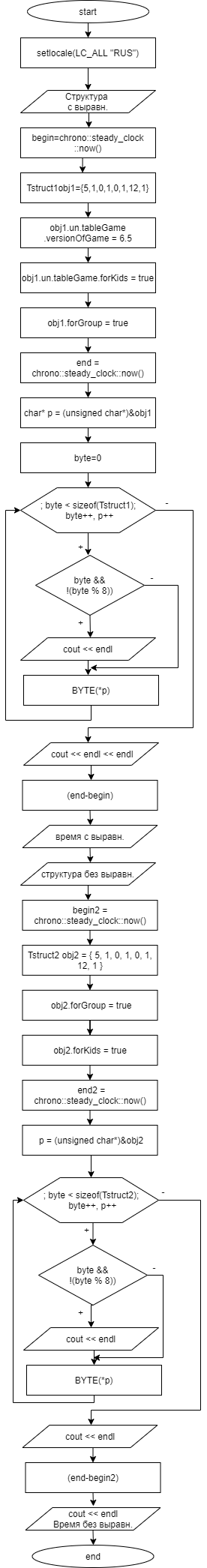


Рис. 4.1 – Перелік властивостей та полів

**Схема алгоритму програми**

****

**Текст програми**

#include <iostream>

#include <chrono>

using namespace std;

void BYTE(unsigned char A) // виведення вмісту байта

{

for (int bit = 128; bit >= 1; bit >>= 1)

cout << (A & bit ? "1" : "0");

cout << " ";

}

struct Tstruct1

{

int s; /\* 4 bytes \*/

short flip1 : 1;

short flip2 : 1;

short flip3 : 1;

short flip4 : 1;

short flip5 : 1;

short flip6 : 4;

short flip7 : 4;

bool forGroup;

union

{

struct

{

bool forKids;

double versionOfGame;

} tableGame;

struct

{

short st;

}computerGame;

} un;

};

struct Tstruct2

{

short s; /\* 4 bytes \*/

int flip1;

int flip2;

int flip3;

int flip4;

int flip5;

int flip6;

int flip7;

bool forGroup;

bool forKids;

};

void main()

{

setlocale(LC\_ALL, "RUS");

cout << "Структура с выравниванием " << endl << endl;

auto begin = chrono::steady\_clock::now();

Tstruct1 obj1 = { 5, 1, 0, 1, 0, 1, 12, 1 };

obj1.un.tableGame.versionOfGame = 6.5;

obj1.un.tableGame.forKids = true;

obj1.forGroup = true;

auto end = chrono::steady\_clock::now();

unsigned char\* p = (unsigned char\*)&obj1;

int byte = 0;

for (; byte < sizeof(Tstruct1); byte++, p++)

{

if (byte && !(byte % 8)) cout << endl;

BYTE(\*p);

}

cout << endl << endl;

auto elapsed\_ms = chrono::duration\_cast<chrono::nanoseconds>(end - begin);

cout << "Время с выравниванием: " << elapsed\_ms.count() << "ns" << endl << endl << endl;

cout << "Структура без выравнивания " << endl << endl;

auto begin2 = chrono::steady\_clock::now();

Tstruct2 obj2 = { 5, 1, 0, 1, 0, 1, 12, 1 };

obj2.forGroup = true;

obj2.forKids = true;

auto end2 = chrono::steady\_clock::now();

p = (unsigned char\*)&obj2;

for (int byte = 0; byte < sizeof(Tstruct2); byte++, p++)

{

if (byte && !(byte % 8)) cout << endl;

BYTE(\*p);

}

cout << endl;

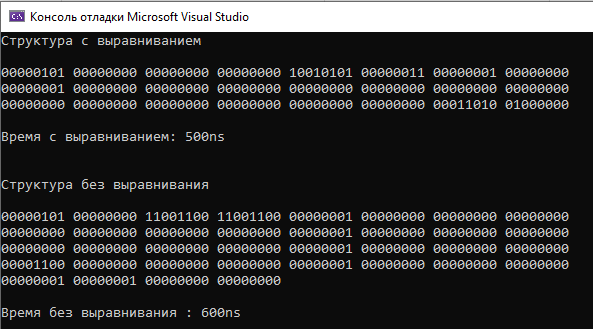
auto elapsed\_ms2 = chrono::duration\_cast<chrono::nanoseconds>(end2 - begin2);

cout << endl;

cout << "Время без выравнивания : " << elapsed\_ms2.count() << "ns" << endl;

}

**Результат роботи програми**

****

Висновок

При виконанні лабораторної роботи, подали внутрішнє (машинне) подання структури з варіантною частиною та з бітовими полями, і масива структур. Структура без вирівнювання затрачує більше пам’яті на зберігання та працює повільніше тому, що якщо елемент не вирівняний, то машина сама буде доповнювати його, щоб прочитати машинні слова, машина буде створювати незначущі байти та зчитувати їх, це сповільнює читання і займає місце в пам’яті.

Контрольні запитання

1. Чим відрізняються машини з порядком Little Endian від машин Big Endian

При порядку **Little\_Endian** бітові поля лунають починаючи з перших байтів, при **Big\_Endian** - навпаки з останніх.

1. Що таке «вірівнювання» і навіщо воно потрібно?

Вирівнювання полягає в тому, щоб адреса, яка вирівнюється, визначалася як числова адреса по модулю ступеня 2. Елемент даних називається вирівняним природньо, якщо його адреса вирівняна по розміру цього елемента. А якщо ні, то – він буде невирівняним. Вирівнювання даних впливає на швидкість виконання програм і на те, чи будуть вони взагалі працювати. Компілятори мови програмування C (та й більшості інших теж) проводять таку оптимізацію: між полями структури вставляються незначні байти для того, щоб у всіх полів було гарне вирівнювання.

1. Як можна змінити вирівнювання даних?

Змінити вирівнювання даних можна за домогою дирректив компілятора.

1. Яким може бути фізичне подання структури?

Структура може бути реалізована на основі масиву з індексом - покажчиком стека і однозв'язного списку.

1. Як розподіляється пам’ять для структур з варіантними частинами?

Під структуру з варіантами виділяється обсяг пам’яті, достатній для розміщення найбільшого варіанта. Якщо ж виділена пам’ять використовується для меншого варіанта, частина її залишається невикористовуваною.

1. Як розподіляється пам’ять для структур з бітовими полями?

Залежно від платформи розташування полів структури в пам'яті може відрізнятися. Зокрема, на Windows порядок розташування наступний: поля на початку структури мають молодші адреси, а поля в кінці структури мають старші адреси.

1. Наведіть приклад доступу до полів структури з варіантної частини.

<ім’я змінної-запису>.<ім’я варіанта>.<ім’я поля

1. Яке призначення дискриптора структури, хто його створює?

Для економії обсягу пам'яті, що відводиться під запис, значення деяких її полів зберігаються в самому дескрипторі, замість покажчиків, тоді в дескрипторі повинні бути записані відповідні ознаки

1. Чи залежить обсяг пам’ті, що розподіляється для структури, від послідовності полів у її складі?

Залежить, , якщо змінити послідовність змінних структури.

1. Які директиви компілятора призначені для впливу на вирівнювання даних?

**#pragma pack (push, <число>)** - задає вирівнювання в <число> байт.

**#pragma pack(pop)** повертає попереднє настроювання.

1. Чи залежить адреса, яку призначає компілятор даним, від типу цих даних?

Кожна змінна С ++ має певний тип, який характеризує розмір і розташування цієї області пам'яті, діапазон значень, які вона може зберігати, і набір операцій, які можна застосувати до цієї змінної.

1. Від чого залежить розмір структури в пам’яті комп’ютера?

Розмір структури в пам’яті залежить від настроювань компілятора та від директив у програмному коді

1. Як вирівнюються структури у масиві структур?

У випадку використання масиву вирівняних структур даних вирівняною в загальному випадку виявляється тільки перша структура масиву. У наслідок чого компілятор додає додаткові невикористовувані байти у кінець кожної оброблюваної структури. Даний процес відбувається таким чином, щоб розмір структури був кратний 2, 4 або 8.