МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
ИНТЕЛЛЕКТУАЛЬНЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

ОТЧЁТ

по лабораторной работе №4

“ Наследование.”

Выполнил:
студент группы ПО-9
Солышко Д.А.

Проверил:
Козик И.Д.

Брест 2023

Лабораторная работа №4
Вариант №19

Цель работы: научиться создавать простейшие классы-наследники.

Основное содержание работы.

Для выполнения лабораторной работы необходимо создать класс Window. В классе должны быть следующие поля: id (int), height (int), width (int), memoryNeeded (int), areAdministatorRightsGranted (boolean) и isShown (boolean). Требуется реализовать конструктор, задающий id и принимающий параметры height, width и areAdministatorRightsGranted, метод doWork, в котором будет изменяться значение memoryNeeded и метод showOrHide, меняющий значение переменной isShown.

Затем необходимо создать класс-наследник класса Window по варианту, реализующий имитацию заданной функциональности.

Работу выполнять на языке C++.

Задание: 4. Создать класс BrowserWindow. В данном классе добавляются поля tabs (массив/список вкладок) и username. Вкладку реализовать отдельным классом. Реализовать методы для добавления вкладок, переключения между ними, закрытия вкладки и поиска по истории вкладок текущего пользователя, а также смену пользователя. MemoryNeeded принимать как количество открытых вкладок плюс величину работы, выполняемой на активной вкладке. В основной программе необходимо реализовать логику использования всех созданных методов.

Текст программы:

```
#include <iostream>
#include <vector>
#include <string>

using namespace std;

class Window {
public:
    int id;
    int height;
    int width;
    int memoryNeeded;
    bool areAdministratorRightsGranted;
    bool isShown;

    Window(int id, int height, int width, bool areAdministratorRightsGranted) {
        this->id = id;
        this->height = height;
        this->width = width;
        this->areAdministratorRightsGranted = areAdministratorRightsGranted;
        this->memoryNeeded = 0;
        this->isShown = false;
    }

    void doWork(int workAmount) {
        memoryNeeded += workAmount;
    }

    void showOrHide() {
        isShown = !isShown;
    }
};

class Tab {
```

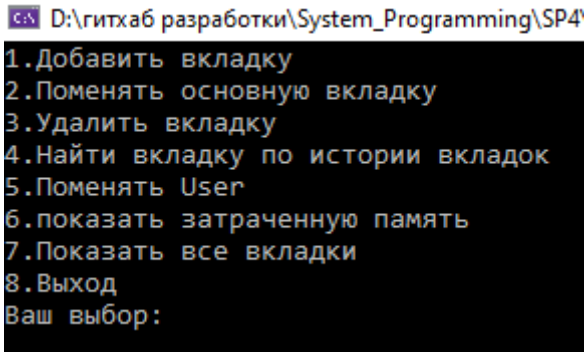
```
public:
    string url;

    Tab(const string url) {
        this->url = url;
    }
};

class BrowserWindow : public Window {
public:
    vector<Tab> tabs;
    string username;
    int currentTabIndex = -1;

    BrowserWindow(int id, int height, int width, bool areAdministratorRightsGranted, string username, int currentTab) :
Window(id, height, width, areAdministratorRightsGranted)
    {
        this->username = username;
        this->currentTabIndex = currentTab;
    }

    BrowserWindow(int id, int height, int width, bool areAdministratorRightsGranted, string username) : Window(id, height,
width, areAdministratorRightsGranted)
    {
        this->username = username;
    }

    void updateMemoryNeeded() {
        memoryNeeded = 0;
        doWork(tabs.size());
        if (currentTabIndex >= 0 && currentTabIndex < tabs.size()) {
            doWork(tabs[currentTabIndex].url.length());
        }
    }

    void addTab(const string url) {
        tabs.emplace_back(url);
        updateMemoryNeeded();
    }

    bool switchTab(int tabIndex) {
        if (tabIndex >= 0 && tabIndex < tabs.size()) {
            if (currentTabIndex != tabIndex) {
                currentTabIndex = tabIndex;
                updateMemoryNeeded();
                cout << "Теперь основная вкладка " << tabs[tabIndex].url << endl;
                return true;
            }
            else {
                cout << "Вы и так на этой вкладке" << endl;
            }
        }
        else {
            cout << "Неправильный индекс" << endl;
        }
        return false;
    }

    bool closeTab(int tabIndex) {
        if (tabIndex >= 0 && tabIndex < tabs.size()) {
            tabs.erase(tabs.begin() + tabIndex);
            cout << "закрытая вкладка : " << tabIndex << endl;
            if (currentTabIndex >= tabIndex)
            {
                currentTabIndex--;
            }
            updateMemoryNeeded();
            return true;
        }
        else {
```

```
            cout << "неправильный индекс" << endl;
        }
        return false;
    }

    void searchHistory(string query) {
        cout << "Поиск результата по " << query << "':" << endl;
        for (int i = 0; i < tabs.size(); ++i) {
            if (tabs[i].url.find(query) != string::npos) {
                cout << "Вкладка " << i + 1 << ": " << tabs[i].url << endl;
            }
        }
    }

    void changeUser(string newUsername) {
        username = newUsername;
    }

    void show()
    {
        cout << "Все вкладки пользователя " << this->username << endl;
        for (int i = 0; i < tabs.size(); i++)
        {
            cout << i + 1 << " " << tabs[i].url << endl;
        }

        if (currentTabIndex != -1)
        {
            cout << "Основная вкладка: " << currentTabIndex + 1 << " " << tabs[currentTabIndex].url << endl;
        }
    }
};

int main() {

    BrowserWindow browser(1, 1024, 768, true, "user1");
    string NameOfTab, NameOfUser;
    int tabIndex = 0;

    setlocale(LC_ALL, "rus");

    while (true)
    {
        int CaseIndex = 0;
        cout << "1.Добавить вкладку" << endl
            << "2.Поменять основную вкладку" << endl
            << "3.Удалить вкладку" << endl
            << "4.Найти вкладку по истории вкладок" << endl
            << "5.Поменять User" << endl
            << "6.показать затраченную память" << endl
            << "7.Показать все вкладки" << endl
            << "8.Выход" << endl
            << "Ваш выбор: " << endl;
        cin >> CaseIndex;
        switch(CaseIndex)
        {
            case 1:
            {
                cout << "Введите URL вкладки, которую хотите добавить ";
                cin >> NameOfTab;
                browser.addTab(NameOfTab);
                break;
            }

            case 2:
            {
                cout << "Введите индекс вкладки, которую хотите сделать основной: ";
                cin >> tabIndex;
                browser.switchTab(--tabIndex);
                break;
```

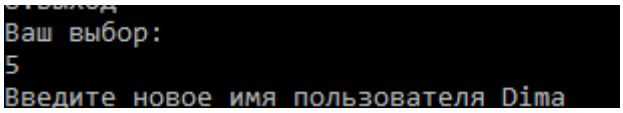
```
        }

        case 3:
        {
           cout << "Введите индекс вкладки, которую хотите удалить ";
           cin >> tabIndex;
           browser.closeTab(--tabIndex);
           break;
        }

        case 4:
        {
           cout << "Введите URL или ключевое имя вкладки, которую хотите найти ";
           cin >> NameOfTab;
           browser.searchHistory(NameOfTab);
           break;
        }

        case 5:
        {
           cout << "Введите новое имя пользователя ";
           cin >> NameOfUser;
           browser.changeUser(NameOfUser);
           break;
        }
        case 6:
        {
           cout << "Затраченная память " << browser.memoryNeeded << endl;
           break;
        }
        case 7:
           browser.show();
           break;
        case 8:
           return 0;
        default:
           break;
     }
   }
}
```

Результаты программы:

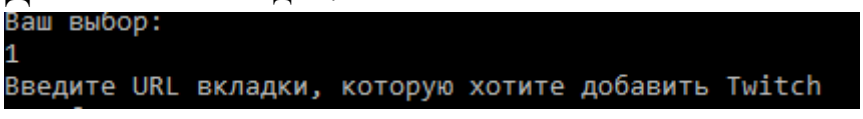
Меню:



Поменяем имя user



Добавление вкладок:



Выставим основную вкладку:

```
Введите индекс вкладки, которую хотите сделать основной: 2
Теперь основная вкладка Twitter
```

Просмотрим все вкладки:

```
Ваш выбор:
7
Все вкладки пользователя Dima
1 Twitch
2 Twitter
3 Youtube
Основная вкладка: 2 Twitter
```

Удалим вкладку Youtube под номером 3

```
Ваш выбор:
3
Введите индекс вкладки, которую хотите удалить 3
```

А после проверим все вкладки

```
Все вкладки пользователя Dima
1 Twitch
2 Twitter
Основная вкладка: 2 Twitter
```

Добавим еще 1 вкладку для того чтобы сделать поиск по части слова Twi

```
Все вкладки пользователя Dima
1 Twitch
2 Twitter
3 Telegram
Основная вкладка: 2 Twitter
```

```
Введите URL или ключевое имя вкладки, которую хотите найти Twi
Поиск результата по Twi':
Вкладка 1: Twitch
Вкладка 2: Twitter
```

Посмотрим затраченную память, если следовать формуле
память = количество вкладок + количество символов в основной вкладке
То результат должен получится 3 + 7 (Twitter) = 10. Все верно

```
Ваш выбор:
6
Затраченная память 10
```

Вывод: я научился создавать простейшие классы-наследники.