МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра прикладной математики и информатики

ОТЧЁТ

по лабораторной работе №2 «Рефакторинг лабораторной работы №1»

по дисциплине

Структуры данных

(наименование дисциплины)

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Санников А.Н.

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Зырянов Е.А.

(подпись) (фамилия, и.,о.)

22-ПМ-1

(шифр группы)

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Задание на лабораторную работу:*** Осуществить рефакторинг лабораторной работы №1 «Списки. Стеки. Очереди».

**Задание проверяемой лабораторной работы**: Имеется n черных и белых карточек, сложенных в стопку. Карточки раскладываются на стол в одну линию следующим образом: первая кладется на стол, вторая под низ стопки, третья- на стол, четвертая - под низ стопки и т.д., пока все карточки не будут выложены на стол. Каким должно быть исходное расположение карточек в стопке, чтобы разложенные на столе карточки чередовались по цвету: белая, черная, белая, черная и т.д.

|  |  |  |
| --- | --- | --- |
| Ошибка | Было | Стало |
| Неправильное комментирование кода | Комментарии отсутствовали | Комментарии упорядочены, код стал хорошо читаем |
| Использование разных стилей кода | Неправильный змеиный регистр | Исправленный змеиный регистр |
| Использование разных стилей расстановки скобок | Отсутствие какого либо стиля расстановки скобок | Применение стиля Кернигена и Ричи |
| Отсутствие пробелов в операторах выражения | В операторах отсутствовали пробелы | Добавлены пробелы |

Код до ревью:

#include <iostream>

#include <fstream>

struct node {

private:

int data;

node\* next;

public:

node() {

next = NULL;

}

int get\_data() {

return data;

}

node\* get\_next() {

return next;

}

void set\_data(int data) {

this->data = data;

}

void set\_next(node\* next) {

this->next = next;

}

int\* get\_data\_link() {

return &data;

}

};

class list {

private:

node\* head = NULL;

public:

void push\_back(int data) {

if (head == NULL) {

head = new node;

head->set\_data(data);

return;

}

node\* tmp = head;

while (tmp->get\_next()) {

tmp = tmp->get\_next();

}

tmp->set\_next(new node);

tmp->get\_next()->set\_data(data);

}

void print() {

node\* tmp = head;

while (tmp) {

std::cout << tmp->get\_data() << " ";

tmp = tmp->get\_next();

}

std::cout << '\n';

}

int\* operator[](size\_t i) {

node\* tmp = head;

for (size\_t j = 0; j < i; j++) {

tmp = tmp->get\_next();

}

return tmp->get\_data\_link();

}

};

int main() {

using namespace std;

setlocale(LC\_ALL, "RUSSIAN");

int n;

cout << "Введите n:";

cin >> n;

int start = 0, step = 2, index = 0, count = 0;

list koloda;

for (size\_t i = 0; i < (2\*n); i++){

koloda.push\_back(0);

}

while (count < 2 \* n) {

index = start;

while (index < 2 \* n) {

\*(koloda[index]) = (count + 1) % 2;

index += step;

count += 1;

}

start = step - 1;

step \*= 2;

}

cout << "Порядок колоды:" << endl;

koloda.print();

return 0;

}

После:

#include <iostream>

// Структура элемента списка

struct Node {

private:

int Data; // Данные элемента

Node\* Next; // Указатель на следующий элемент списка

public:

Node() { // Конструктор по умолчанию

Next = nullptr; // Инициализация указателя на следующий элемент значением nullptr

}

// Получить данные элемента

int Get\_Data() {

return Data; // Возвращает данные элемента

}

// Получить указатель на следующий элемент

Node\* Get\_Next() {

return Next; // Возвращает указатель на следующий элемент

}

// Установить данные элемента

void Set\_Data( int data ) {

Data = data; // Присваивает переданные данные

}

// Установить указатель на следующий элемент

void Set\_Next( Node\* next ) {

Next = next; // Присваивает переданный указатель

}

// Получить ссылку на данные элемента

int\* Get\_Data\_Link() {

return &Data; // Возвращает ссылку на данные элемента

}

};

// Класс списка

class Linked\_List {

private:

Node\* Head = nullptr; // Указатель на голову списка

public:

// Добавить элемент в конец списка

void Push\_Back( int data ) {

if ( Head == nullptr ) {

Head = new Node; // Создание нового узла списка

Head→Set\_Data( data ); // Установка данных в новом узле

return;

}

Node\* tmp = Head;

while ( tmp→Get\_Next() ) {

tmp = tmp->Get\_Next(); // Переход к следующему узлу

}

tmp→Set\_Next( new Node ); // Создание нового узла списка после текущего узла

tmp→Get\_Next()→Set\_Data( data ); // Установка данных в новом узле

}

// Вывести элементы списка

void Print() {

Node\* tmp = Head;

while ( tmp ) {

std::cout << tmp->Get\_Data() << " "; // Вывод данных текущего узла

tmp = tmp->Get\_Next(); // Переход к следующему узлу

}

std::cout << '\n';

}

// Получить элемент списка по индексу

int\* operator[]( size\_t index ) {

Node\* tmp = Head;

for ( size\_t i = 0; i < index; i++ ) {

tmp = tmp->Get\_Next(); // Переход к следующему узлу

}

return tmp->Get\_Data\_Link(); // Возврат ссылки на данные текущего узла

}

};

int main() {

using namespace std;

setlocale( LC\_ALL, "RUSSIAN" );

int N; // Количество элементов

cout << "Введите количество элементов:";

cin >> N;

Linked\_List List; // Создание списка

// Заполнение списка

for ( size\_t i = 0; i < N; i++ ) {

int data;

cout << "Введите элемент " << i + 1 << ":";

cin >> data;

List.Push\_Back( data ); // Добавление элемента в список

}

cout << "Элементы списка:";

List.Print(); // Вывод списка

return 0;

}