ДЗ-1. Срок сдачи — до 27.02.21

1. Написать класс с данными разных типов, среди которых обязательно должны быть указатели.

Для этого класса написать конструкторы (больше одного + конструктор копирования), деструктор, оператор присваивания и хотя бы одну произвольную функцию-член класса. В каждую из специальных функций класса включить отладочный вывод на экран, чтобы было понятно, какая функция работает.

Функция main должна демонстрировать работу с объектами данного класса.

Использование STL и типа string запрещено.

ДЗ-2. (задание практикума №1, часть 1) Срок сдачи — до 06.03.21

1. АТД. Список данных.

Класс — список (двунаправленный), элемент данных — int (либо произвольного типа). Смотрите ДЗ-3, этот класс «Список» потребуется для дальнейшей работы, поэтому желательно, чтобы его легко было потом адаптировать для работы с данными произвольного типа.

Определить необходимые конструкторы.
Обеспечить корректное уничтожение объектов.

В классе должны быть функции:
добавления элемента в начало (**push_front**) и в конец (**push_back**),
чтение первого элемента списка (**front**),
чтение последнего элемента списка (**back**),
удаление первого элемента списка (**pop_front**),
удаление последнего элемента списка (**pop_back**),
добавление элемента x перед позицией p (**insert(p,x)**),
удаление элемента из позиции p (**erase(p)**)
проверка списка на пустоту (**empty**),
текущее число элементов (**size**),
вывод информации об элементах списка (**print**).

Использование STL и типа string запрещено.

Функция main должна демонстрировать работу с объектами данного класса.
Пример работы: List l1; l1.push_front(1); ... l1.print(); и т.д.

ДЗ-3. (задание практикума №1, часть 2) Срок сдачи — до 13.03.21

2. АТД. Очередь для хранения данных.

На основе класса Список_элементов_произвольного_типа (использовать класс из ДЗ-2)

определить класс **Очередь**, который **должен быть производным от Списка**.
Элемент данных, помещаемых в очередь — см. вариант задания.
№ варианта = № в списке группы % 3 +1 .

Максимальный допустимый размер очереди определен по умолчанию, а также может явно указываться при создании объекта-Очереди.

Определить необходимые конструкторы.
Обеспечить корректное уничтожение объектов.

В классе должны быть функции:
добавления элемента в конец очереди (**back**),
чтение первого элемента из очереди без его удаления (**front**) ,
удаление первого элемента очереди (**pop**),
проверка очереди на пустоту (**empty**),
текущее число элементов (**size**),
проверка, что очередь целиком заполнена (**full**).
вывод информации об элементах очереди без ее изменения (**print**).

Замечания:
Хотелось бы, чтобы класс список можно было использовать для хранения любых данных. Поэтому нужно выделить данные в отдельный класс.
Функции класса Список должны остаться прежними, а не переделываться под потребности очереди.
Через объект Очередь должны быть доступны только функции, указанные в задании. При этом базовый класс остается полноценным списком.
Глобальных переменных быть НЕ должно.

Использование STL и типа string запрещено.

Пример работы: `Queue q1(5), q2; List l1; q1.back(el); l1.push_back(1); l1.print(); cout<<q2.size();` и т.д.

Варианты данных, которые помещаются в очередь.

1. **Элемент данных -** объект, содержащий информацию о клиенте: фамилия, время добавления в очередь (целое или строка). Хранится также информация о текущем количестве клиентов в очереди и об общем количестве клиентов во всех очередях (списках).

2. **Элемент данных** – объект, содержащий информацию о заказе: название фирмы, номер телефона (целое или строка), номер заказа. Нумерация заказов единая для всех списков или очередей.

3. **Элемент данных –** объект «банковский счет». Необходимые члены-данные: номер счета, владелец счета, дата создания счета (число или строка), сумма денег,

которая на нем хранится. Нумерация счетов единая для всех очередей (списков).

ДЗ-4. Абстрактные классы. Срок сдачи — до 20.03.21

Написать абстрактный класс (содержание произвольное) и несколько (>=2) производных от него классов. Определить независимый от этой иерархии класс, который работает с массивом объектов типа абстрактного класса.
Использование STL запрещено.
Функция main должна демонстрировать работу с объектами указанных классов.

ДЗ-5. Перегрузка операций. Срок сдачи — до 27.03.21

Написать (кто уже писал – дополнить) класс Матрица таким образом, чтобы к объектам этого типа была применима двойная индексация.

Должен быть верным, например, следующий фрагмент программы:

```
Matrix m; … m[1][2] = 5;  int x = m[2][3];
const Matrix mc;   cout<< mc[1][1];
// mc[1][1] =100 ; // ошибка компиляции
```