Задача 2

Реализация

В классе будут храниться текущее число элементов в деке ($size$), массив размера $capacity$ и индекс первого элемента дека $iFirst$. Изначально $capacity$ равно единице. В каждый момент времени (если дек не пустой) будет выполняться

$$size \leqslant capacity \leqslant size * 4$$

Элементы дека будут находиться (по порядку, начиная с первого) в массиве на следующих индексах: $[iFirst, iFisrt + size)$, если $iFisrt + size \leqslant capacity$ и на $[iFirst, capacity)$ и $[0, size - (capacity - iFirst))$ иначе.

- **push_back, push_front**
  Если $size \neq capacity$, то записываем в соответствующую $((iFirst - 1 + capacity)\ mod\ capacity$ в случае pop и $(iFirst + size)\ mod\ capacity$ в случае push) ячейку массива элемент, увеличиваем значение $size$ на единицу и обновляем $iFirst$ если нужно (то есть в случае операции $push\_front$).

  Иначе создаём новый массив размера $capacity * 2$, копируем соответствующие $size$ элементов из старого массива в новый (на первые $size$ позиций), делаем присваивание $iFirst = 0$ старый массив удаляем, далее делаем аналогично случаю $size \neq capacity$.

- **pop_back, pop_front**
  Вначале уменьшим значение $size$ на единицу и обновим значение $iFirst$ если нужно (то есть в случае операции pop_front). Далее, если $capacity > size*4$, то создаём новый массив размера $\frac{capacity}{2}$, копируем соответствующие $size$ элементов из старого массива в новый (на первые $size$ позиций), старый массив удаляем и делаем присваивание $iFirst = 0$.

- **Доступ по идексу**
  Возвращаем соответствующий элемент массива.

Амортизационный анализ

Полностью (символ в символ) повторяет таковой для вектора, поэтому я не стал копировать сюда то что было в векторе.