**2 Вариант.** Знакомство с контейнерами STL. Приемы управления памятью. Изучение поведения умных указателей shared\_ptr.

В ходе выполнения домашнего задания я углубил свои знания языка программирования C++.

**MemoryTrick**:

Класс vector вычисляет, на какое значение увеличить вместимость, в зависимости от текущей вместимости.

Пусть Nold – старая вместимость, Nnew – новая. Очевидно, что новая вместимость должна быть больше текущей, т.е. Nnew > Nold, так как в «новую» необходимо скопировать все текущие данные. Также в новой вместимости нужно учесть место для у.е. новых данных. Получаем: Nnew = Nold + 1.

Но если заново выделять новую память при каждом новом элементе, то, вероятно, программа будет работать медленно. Особенно это заметно на больших значениях вместимости. Например, при 10000. Таким образом, при добавлении всего лишь одного элемента, придётся копировать все предыдущие 10000 элементов.

Однако, если взять любое значение вместимости и разделить на предыдущее, то мы получим число, приблизительно равное 1.5. Такой подход позволяет более эффективно работать с выделением памяти и не копировать данные при каждом добавлении элемента.

Оценка эффективности при добавлении 1000 элементов:

- При «наивной» реализации потребуется выделять память 1000 раз, то есть 1000 раз копировать все предыдущие значения. То есть 1000 раз вызывать конструктор и деструктор. Если учесть, что vector может хранить пользовательские классы, которые могут содержать значительное количество данных, то такая реализация положит конец производительности программы;

- При «умной» реализации будет произведено 18 выделений памяти, что в разы эффективнее.

**Умные указатели**:

Позволяют эффективнее работать с памятью за счёт того, что теперь динамически выделенную память освобождает не человек, а программа

Как работают умные указатели? В C++ объект живёт до тех пор, пока не закончится блок кода, в котором он был объявлен (знак }). Разумеется, это правило не распространяется на динамически-выделенную память, она освобождается командой delete (delete[]). Умные указатели уничтожаются вместе с тем, как заканчивается блок кода, при этом умный указатель вызывает деструктор у объекта, на который указывает.

Таким образом, отпадает необходимость следить за утечками динамической памяти. Однако, если место нужно «здесь и сейчас» (то есть, не дожидаясь конца блока), то придётся использовать обычные указатели и следить за утечками.

«Общий размер отчета не должен превышать 1 стр.»? :(