**ЗВІТ ПРО ВИКОНАНЕ ЗАВДАННЯ**

по курсу "Методи верифікації і оптимізації програм"

студента групи ПК-16м-1

Бекленищева Владислава Ігоровича

ЗАВДАННЯ № 1

кафедра комп’ютерних технологій, ДНУ

2016/2017 навч. р.

# 

# Постановка задачи

Вариант - 1

Даны натуральные числа: N, a[1], a[2], ..., a[N]. Определить количество членов a[K] последовательности a[1], ..., a[N], имеющих четные порядковые номера и являющихся нечетными числами.

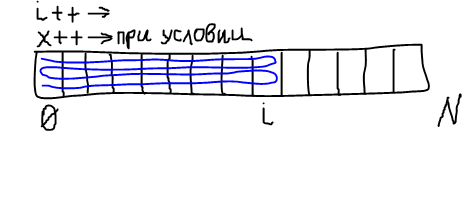
# Описание решения задачи

Основная идея решения задачи состоит в том чтобы пройтись по всем элементам массива от начала до конца и подсчитать количество элементов, которые стоят на четных порядковых номерах (even) и одновременно являются нечетными числами (odd). Проверку на четность / нечетность можно выполнять с помощью операции “взятии остатка от деления” (%), однако в доказательстве вместо выражений с этой операцией используются соответствующие предикаты even и odd. Например, следующий образом проверяется нечетность элемента, стоящего на позиции i: .

Основной стартовой точкой для доказательство является построение спецификации программы. Так как программа является циклом с предварительной инициализацией, кроме предикатов предусловия Q, программы S и предиката постусловия R, нужно еще составить предикаты I, R и t, которые являются командой инициализации, инвариантом и ограничивающей функцией соответственно.

Для удобства предположим, что индексация массива производится от 0 до N-1, а N может равняться или быть больше 0 (например, когда массив пустой). Сразу отсюда следует предусловие Q: .

Далее нужно найти инвариант. Инвариант - предикат, истинность которого не должна нарушаться в ходе работы цикла и перед его началом. Это понятие является довольно важным в теории верификации программ и самом программировании, так как при разработке цикла нужно как раз реализовать его инвариант. В данной задачи, инвариант может выглядеть следующим образом:



На рисунке синим отмечена область, которую мы уже прошли в процессе поиска. Верхняя граница синей области имеет значение i. Таким образом, в данной задаче инвариант состоит из двух условий:

* Указатель на элемент нужно двигать от начала массива (от 0) в конец (до N), передвигая его на 1 элемент вперед на каждой итерации (.
* При нахождении нечетного элемента, стоящего на четном месте (проверяем i), стоит увеличить счетчик количества на 1 (.

Для нахождения количества элементов в теории доказательства правильности программ есть квантор количества. Таким образом, инвариант в предикатной форме можно подать в таком виде:

С инварианта можно получить постусловие, то есть предикат R. Постусловием является то, что должно быть результатом работы программы. В данной задачи нас интересует количество чисел, которое возвращает квантор количества:

Кроме инварианта еще одним важным предикатом, является целочисленная неотрицательная, ограничивающая функция t (bound t). Во время работы цикла она должна уменьшаться и приближаться к 0 (нижняя граница). В данной задачи t однозначно зависит от числа шагов, которые осталось выполнить, то есть t будет равна: . Таким образом, на каждой итерации t уменьшается в то время, как i увеличивается на 1.

Команда инициализации может быть построена из инварианта P и предусловия Q. В данном случае она будет следующей: То есть указатель i показывает на первый элемент массива, а x, то есть количество, в самом начале равно 0.

Охрана цикла записывается исходя из ограничивающей функции, то есть N - i > 0. Сама программа S записывается для данной задачи в виде IF конструкции: если элемент на четной позиции и нечетный, то увеличиваем счетчик x и i, а есть условие не выполняется, то увеличиваем только i. Всю команду повторения можно записать в таком виде:

Само доказательство (вместе со спецификацией) программы с комментариями описано в следующем пункте отчета.

# Спецификация и доказательство программы

// выполним подстановку:

// является

// а в кванторе количества дает , а значит квантор возвращает 0

// 0 = 0 является TRUE:

/\*-----------------------------------------------------------

\*

\* // по закону импликации

\* // по закону исключения третьего

\*----------------------------------------------------------\*/

// по

// возьмём консеквент импликации

// по

//закону исключения третьего

// тоже является :

// распишем выражение:

// выполняем команду подстановки в предикат P:

// из первого квантора выносим 1 элемент и изменяем область j на

// из второго квантора выносим 0 элемент и изменяем область j на , так как

// элемент посылки не является , тем который нам нужен

// сокращаем единицы в уравнении с первым квантором

// и получим, что правые части импликаций равны:

// обозначим левую часть через , а правую через и докажем

/\*--------------------------------------------------------------------

\*

\* // по закону импликации

\* // дистрибутивность дизъюнкции

\* // по закону противоречия

\*

\* // по закону упрощения дизъюнкции

\*---------------------------------------------------------------------\*/

// по останется только правая часть импликации:

// вернемся к исходному выражению

// объединим и , получим:

// правую часть распишем по закону дистрибутивности конъюнкции:

// получим, что один из дизьюнктов консеквента однозначно равен антецеденту выражения

// и поэтому по НЛЛ 2 мы получаем:

// объединим и , получим:

// по НЛЛ 1:

// объединим и , получим:

// получаем, что один из конъюнктов левой части однозначно совпадает с

// консеквентом импликации, значит применяем НЛЛ 1:

// распишем консеквент импликации

// по команде последовательности получим:

// распишем выражение, в которое будем выполнять подстановку

// первые 2 конъюнкта и становятся и мы получим:

// выполняем подстановку:

// получаем, что правые части импликаций равны

// воспользуемся ранее доказанной леммой и получим:

// вернемся к исходному выражению

// выполняем подстановку

// консеквент является :

// докажем

/\*------------------------------------------------------

\*

\*

\* // по закону упрощения дизъюнкции

\*

\*-------------------------------------------------------\*/

// окончательно получим по