Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра вычислительных систем

**ОТЧЕТ**

по практической работе 3

по дисциплине «**Программирование**»

|  |  |  |
| --- | --- | --- |
| Выполнил:  студент гр. ИС-241  «17» марта 2023 г. | \_\_\_\_\_\_\_\_\_\_\_\_ | /Дмитрюк В.В./ |
|  |  |  |
| Проверил:  ст.преп. Кафедры ВС  «\_\_» марта 2023 г. | \_\_\_\_\_\_\_\_\_\_\_\_ | /Фульман В.О./ |

Оценка «\_\_\_\_\_\_\_\_\_\_\_\_\_»

Новосибирск 2023

**ОГЛАВЛЕНИЕ**

[ЗАДАНИЕ 3](#_Toc35593781)

[ВЫПОЛНЕНИЕ РАБОТЫ 4](#_Toc35593782)

[ПРИЛОЖЕНИЕ 5](#_Toc35593783)

# **ЗАДАНИЕ**

*Реализовать тип данных «Динамический массив целых чисел» – IntVector и основные функции для работы с ним. Разработать тестовое приложение для демонстрации реализованных функций.*

**ВЫПОЛНЕНИЕ РАБОТЫ**

Прототипы необходимых для реализации функций даны в требованиях к работе и были записаны в заголовочный файл IntVector.h. Функции были реализованы в файле IntVector.c.

Реализация функций происходила в порядке, отличном от того, в котором даны их прототипы – это позволяет использовать раннее реализованные функции в последующих.

Структура вектора выглядит следующим образом:

|  |  |
| --- | --- |
| 1  2  3  4  5 | **typedef** **struct** intVector {  **size\_t** size; // длина массива, она же размер **size\_t** cap; // ёмкость массива  **int**\* arr; // массив чисел  } IntVector; |

**IntVector \*int\_vector\_new(size\_t initial\_capacity)**

Реализация: выделение памяти под структуру и массив целых чисел данной ёмкости с проверкой на нулевой указатель.

**IntVector \*int\_vector\_copy(const IntVector \*v)**

Реализация: создаёт новый вектор с ёмкостью, равной ёмкости данного вектора при помощи предыдущей функции, результат работы которой проверяется, и копирует массив целых чисел вместе со значением размера в новый вектор.

**void int\_vector\_free(IntVector \*v)**

Реализация: освобождает память массива целых чисел, а затем – память всей структуры.

|  |  |
| --- | --- |
| 1 | **static** **void** **\_\_throw\_index\_OOB\_error**(**size\_t** index, **size\_t** size) |

Эта функция – следствие попытки решить задачу «Advanced». Однако, задача решена не полностью – функция не предоставляет возможности обработать ошибку в пользовательском коде, а аварийно завершает программу при попытке обратиться к индексу за пределами размера вектора.

**int int\_vector\_get\_item(const IntVector \*v, size\_t index)**

Реализация: возвращает элемент массива целых чисел с запрошенным индексом, осуществляя проверку индекса на принадлежность пределу (0..size), вызывая предыдущую функцию при выходе за пределы.

**void int\_vector\_set\_item(IntVector \*v, size\_t index, int item)**

Реализация: аналогично предыдущей функции, но вместо чтения осуществляется запись значения item.

**size\_t int\_vector\_get\_size(const IntVector \*v)**

Реализация: возвращает значение поля size.

**size\_t int\_vector\_get\_capacity(const IntVector \*v)**

Реализация: возвращает значение поля cap.

**int int\_vector\_reserve(IntVector \*v, size\_t new\_capacity)**

Реализация: при помощи функции realloc, возвращаемое значение которой проверяется, под массив целых чисел выделяется больше памяти, а значение ёмкости увеличивается. При попытке уменьшить размер массива этой функцией не происходит ничего.

**int int\_vector\_resize(IntVector \*v, size\_t new\_size)**

Реализация: если новый размер не укладывается в ёмкость, вызывается предыдущая функция для её увеличения, после чего происходит инициализация новых элементов нулём и изменение значения размера в структуре.

**int int\_vector\_push\_back(IntVector \*v, int item)**

Реализация: записывает в массив целых чисел элемент item по индексу size. Если такая запись приведёт к выходу за ёмкость массива, ёмкость увеличивается функцией **int\_vector\_reserve** до 2n+1, где n – старое значение ёмкости. Это отходит от рекомендуемого коэффициента роста и от реализации вектора в C++, поскольку позволяет успешно добавить элемент даже при нулевом размере.

**void int\_vector\_pop\_back(IntVector \*v)**

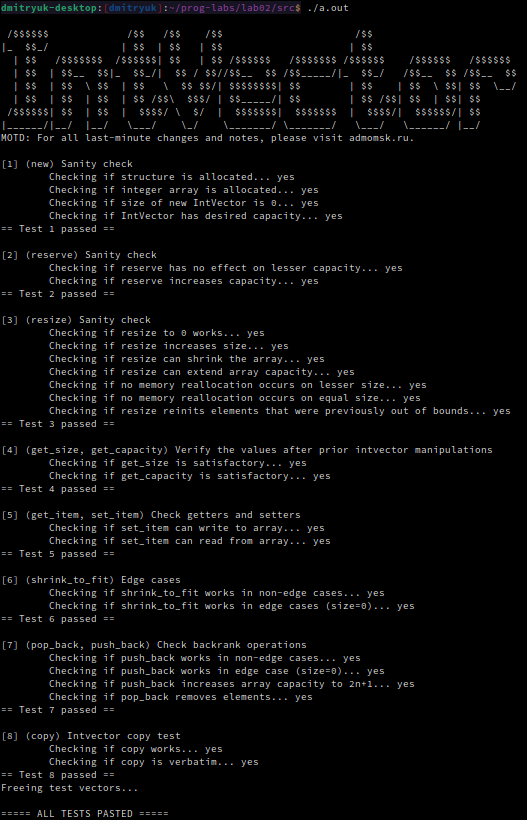
Реализация: записывает 0 в элемент под индексом size и декрементирует размер.

**int int\_vector\_shrink\_to\_fit(IntVector \*v)**

Реализация: при помощи realloc выделяет минимальное количество памяти для хранения количества целых чисел, соответствующего размеру. Значение возврата realloc проверяется. Если размер нулевой, память массива целых чисел освобождается. Если размер равен текущей ёмкости, не происходит ничего.

В ходе работы была реализована тестовая программа, проверяющая правильность работы всех реализованных функций. Значения после вызова функций сравниваются со значениями, известными на этапе компиляции и соответствующие ожидаемым. Программа выводит результаты тестов в стандартный поток ввода-вывода и уведомляет пользователя о прохождении всех тестов. Если какое-то из значений не соответствует ожидаемым, тест считается проваленным и работа тестовой программы прерывается.

Для удобства сборки тестовой программы был написан Makefile.



1. Снимок экрана после выполнения тестовой программы

# **ПРИЛОЖЕНИЕ**

Исходный код с комментариями;

**example.c**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | #include <assert.h>  #include <stddef.h>  #include <stdint.h>  **size\_t** **encode\_varint**(**uint32\_t** value, **uint8\_t**\* buf)  {  assert(buf != NULL);  **uint8\_t**\* cur = buf;  **while** (value >= **0x80**) {  **const** **uint8\_t** byte = (value & **0x7f**) | **0x80**;  \*cur = byte;  value >>= **7**;  ++cur;  }  \*cur = value;  ++cur;  **return** cur - buf;  }  **uint32\_t** **decode\_varint**(**const** **uint8\_t**\*\* bufp)  {  **const** **uint8\_t**\* cur = \*bufp;  **uint8\_t** byte = \*cur++;  **uint32\_t** value = byte & **0x7f**;  **size\_t** shift = **7**;  **while** (byte >= **0x80**) {  byte = \*cur++;  value += (byte & **0x7f**) << shift;  shift += **7**;  }  \*bufp = cur;  **return** value;  } |