МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №14**

по дисциплине: Основы программирования

тема: «Реализация структуры данных «Вектор»»

Выполнил: ст. группы ПВ-223

Пахомов Владислав Андреевич

Проверили:

Притчин Иван Сергеевич

Черников Сергей Викторович

Код-ревьер: ст. группы ПВ-223

Голуцкий Георгий Юрьевич

Белгород 2023 г.

**Лабораторная работа № 14**

**Содержание отчёта:**

* Тема лабораторной работы.
* Цель лабораторной работы.
* Ссылка на открытый репозиторий с решением.
* Исходный код файлов
  + vector.h / vector.c
  + \*\*vectorVoid.h / vectorVoid.c
  + main.c
* Результат выполнения команд

git log --stat -- libs/data\_structures/vector/ main.c

\*git log --stat -- libs/data\_structures/vectorVoid/

* Вывод по работе.

**Тема лабораторной работы:** Реализация структуры данных «Вектор».

**Цель лабораторной работы:** усовершенствование навыков в создании библиотек, получение навыков работы с системой контроля версий 𝑔𝑖𝑡.

**Ссылка на репозиторий:**

<https://github.com/IAmProgrammist/programming-and-algorithmization-basics/tree/c-lab14>

**Исходный код файлов:**

vector.h

#ifndef PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTOR\_H  
#define PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTOR\_H  
  
#include <stdint.h>  
#include <stdlib.h>  
#include <stdio.h>  
#include <stdbool.h>  
  
#define min(a, b) (a) < (b) ? (a) : (b)  
#define max(a, b) (a) > (b) ? (a) : (b)  
  
typedef struct Vector {  
 int \*data;  
 size\_t size;  
 size\_t capacity;  
} Vector;  
  
Vector createVector(size\_t n);  
  
void reserve(Vector \*v, size\_t newCapacity);  
  
void clear(Vector \*v);  
  
void shrinkToFit(Vector \*v);  
  
void deleteVector(Vector \*v);  
  
bool isEmpty(Vector \*v);  
  
bool isFull(Vector \*v);  
  
int getVectorValue(Vector \*v, size\_t i);  
  
void pushBack(Vector \*v, int x);  
  
void popBack(Vector \*v);  
  
int\* atVector(Vector \*v, size\_t index);  
  
int\* back(Vector \*v);  
  
int\* front(Vector \*v);  
  
#endif *//PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTOR\_H*

vector.c

#include "vector.h"  
  
Vector createVector(size\_t n) {  
 Vector resultVector = {malloc(sizeof(int) \* n), 0, n};  
  
 if (resultVector.data == **NULL**) {  
 fprintf(**stderr**, "bad alloc");  
 exit(1);  
 }  
 return resultVector;  
}  
  
void reserve(Vector \*v, size\_t newCapacity) {  
 if (newCapacity == 0) {  
 deleteVector(v);  
 \*v = (Vector) {**NULL**, 0, 0};  
 } else {  
 \*v = (Vector) {realloc(v->data, newCapacity \* sizeof(int)), **min**(newCapacity, v->size), newCapacity};  
  
 if (v->data == **NULL**) {  
 fprintf(**stderr**, "bad alloc");  
 exit(1);  
 }  
 }  
}  
  
void clear(Vector \*v) {  
 v->size = 0;  
}  
  
void shrinkToFit(Vector \*v) {  
 reserve(v, v->size);  
}  
  
void deleteVector(Vector \*v) {  
 free(v->data);  
}  
  
**bool** isEmpty(Vector \*v) {  
 return v->size == 0;  
}  
  
**bool** isFull(Vector \*v) {  
 return v->size == v->capacity;  
}  
  
int getVectorValue(Vector \*v, size\_t i) {  
 return v->data[i];  
}  
  
void pushBack(Vector \*v, int x) {  
 if (isFull(v))  
 reserve(v, **max**(1, v->capacity \* 2));  
  
 v->data[v->size++] = x;  
}  
  
void popBack(Vector \*v) {  
 if (isEmpty(v)) {  
 fprintf(**stderr**, "can't pop element from an empty array");  
 exit(1);  
 }  
  
 v->size--;  
}  
  
int\* atVector(Vector \*v, size\_t index) {  
 if (index < v->size)  
 return v->data + index;  
  
 fprintf(**stderr**, "IndexError: a[%zu] is not exists", index);  
  
 return **NULL**;  
}  
  
int\* back(Vector \*v) {  
 return atVector(v, v->size - 1);  
}  
  
int\* front(Vector \*v) {  
 return atVector(v, 0);  
}

vectorVoid.h

#ifndef **PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTORVOID\_H**#define **PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTORVOID\_H**#include <stdint.h>  
#include <stddef.h>  
#include <stdlib.h>  
#include <stdio.h>  
#include <stdbool.h>  
#include <string.h>  
  
#define **min**(a, b) (a) < (b) ? (a) : (b)  
#define **max**(a, b) (a) > (b) ? (a) : (b)  
  
typedef struct VectorVoid {  
 void \*data;  
 size\_t size;  
 size\_t capacity;  
 size\_t baseTypeSize;  
} VectorVoid;  
  
VectorVoid createVectorV(size\_t n, size\_t baseTypeSize);  
  
void reserveV(VectorVoid \*v, size\_t newCapacity);  
  
void shrinkToFitV(VectorVoid \*v);  
  
void clearV(VectorVoid \*v);  
  
void deleteVectorV(VectorVoid \*v);  
  
**bool** isEmptyV(VectorVoid \*v);  
  
**bool** isFullV(VectorVoid \*v);  
  
void getVectorValueV(VectorVoid \*v, size\_t index, void \*destination);  
  
void setVectorValueV(VectorVoid \*v, size\_t index, void \*source);  
  
void popBackV(VectorVoid \*v);  
  
void pushBackV(VectorVoid \*v, void \*source);  
  
#endif *//PROGRAMMING\_AND\_ALGORITHMIZATION\_BASICS\_VECTORVOID\_H*

vectorVoid.c

#include "vectorVoid.h"  
  
VectorVoid createVectorV(size\_t n, size\_t baseTypeSize) {  
 VectorVoid resultVector = {malloc(baseTypeSize \* n), 0, n, baseTypeSize};  
  
 if (resultVector.data == **NULL**) {  
 fprintf(**stderr**, "bad alloc");  
 exit(1);  
 }  
  
 return resultVector;  
}  
  
void reserveV(VectorVoid \*v, size\_t newCapacity) {  
 if (newCapacity == 0) {  
 deleteVectorV(v);  
  
 \*v = (VectorVoid) {**NULL**, 0, 0, v->baseTypeSize};  
 } else {  
 \*v = (VectorVoid) {realloc(v->data, newCapacity \* v->baseTypeSize), **min**(newCapacity, v->size), newCapacity,  
 v->baseTypeSize};  
  
 if (v->data == **NULL**) {  
 fprintf(**stderr**, "bad alloc");  
 exit(1);  
 }  
 }  
}  
  
void shrinkToFitV(VectorVoid \*v) {  
 reserveV(v, v->size);  
}  
  
void clearV(VectorVoid \*v) {  
 v->size = 0;  
}  
  
void deleteVectorV(VectorVoid \*v) {  
 free(v->data);  
}  
  
**bool** isEmptyV(VectorVoid \*v) {  
 return v->size == 0;  
}  
  
**bool** isFullV(VectorVoid \*v) {  
 return v->size == v->capacity;  
}  
  
void getVectorValueV(VectorVoid \*v, size\_t index, void \*destination) {  
 char \*source = (char \*) v->data + index \* v->baseTypeSize;  
 **memcpy**(destination, source, v->baseTypeSize);  
}  
  
void setVectorValueV(VectorVoid \*v, size\_t index, void \*source) {  
 char \*destination = (char \*) v->data + index \* v->baseTypeSize;  
 **memcpy**(destination, source, v->baseTypeSize);  
}  
  
void popBackV(VectorVoid \*v) {  
 if (isEmptyV(v)) {  
 fprintf(**stderr**, "can't pop element from an empty array");  
 exit(1);  
 }  
  
 v->size--;  
}  
  
void pushBackV(VectorVoid \*v, void \*source) {  
 if (isFullV(v))  
 reserveV(v, **max**(1, v->capacity \* 2));  
  
 setVectorValueV(v, v->size++, source);  
}

main.c

#include <assert.h>  
  
#include "libs/data\_structures/vector/vector.h"  
#include "libs/data\_structures/vector/vectorVoid.h"  
  
void test\_pushBack\_emptyVector() {  
 Vector r = createVector(0);  
  
 pushBack(&r, 42);  
 **assert**(getVectorValue(&r, 0) == 42 && r.size == 1 && r.capacity == 1);  
}  
  
void test\_pushBack\_fullVector() {  
 Vector r = createVector(3);  
  
 *// Filling vector* pushBack(&r, 421);  
 pushBack(&r, 422);  
 pushBack(&r, 423);  
  
 pushBack(&r, 42);  
  
  
 **assert**(getVectorValue(&r, 3) == 42 && r.size == 4 && r.capacity == 6);  
  
 deleteVector(&r);  
  
 r = createVector(0);  
  
 pushBack(&r, 1);  
 **assert**(getVectorValue(&r, 0) == 1 && r.size == 1 && r.capacity == 1);  
  
 pushBack(&r, 6);  
 **assert**(getVectorValue(&r, 1) == 6 && r.size == 2 && r.capacity == 2);  
  
 pushBack(&r, 5);  
 **assert**(getVectorValue(&r, 2) == 5 && r.size == 3 && r.capacity == 4);  
  
 pushBack(&r, 7);  
 **assert**(getVectorValue(&r, 3) == 7 && r.size == 4 && r.capacity == 4);  
  
 pushBack(&r, 42);  
 **assert**(getVectorValue(&r, 4) == 42 && r.size == 5 && r.capacity == 8);  
}  
  
void test\_atVector\_notEmptyVector() {  
 Vector r = createVector(0);  
  
 pushBack(&r, 1);  
 pushBack(&r, 6);  
 pushBack(&r, 5);  
 pushBack(&r, 7);  
 pushBack(&r, 42);  
  
 **assert**(\*atVector(&r, 1) == 6);  
 **assert**(\*atVector(&r, 2) == 5);  
 **assert**(\*atVector(&r, 3) == 7);  
}  
  
void test\_atVector\_requestToLastElement() {  
 Vector r = createVector(0);  
  
 pushBack(&r, 1);  
 pushBack(&r, 6);  
 pushBack(&r, 5);  
 pushBack(&r, 7);  
 pushBack(&r, 42);  
  
 **assert**(\*atVector(&r, 4) == 42);  
}  
  
void test\_back\_oneElementInVector() {  
 Vector r = createVector(0);  
  
 pushBack(&r, 42);  
  
 **assert**(\*back(&r) == 42);  
}  
  
void test\_front\_oneElementInVector() {  
 Vector r = createVector(0);  
  
 pushBack(&r, 42);  
  
 **assert**(\*front(&r) == 42);  
}  
  
void test\_popBack\_notEmptyVector() {  
 Vector v = createVector(0);  
 pushBack(&v, 10);  
 **assert**(v.size == 1);  
 popBack(&v);  
 **assert**(v.size == 0);  
 **assert**(v.capacity == 1);  
}  
  
void test() {  
 test\_pushBack\_emptyVector();  
 test\_pushBack\_fullVector();  
 test\_popBack\_notEmptyVector();  
 test\_atVector\_notEmptyVector();  
 test\_atVector\_requestToLastElement();  
 test\_back\_oneElementInVector();  
 test\_front\_oneElementInVector();  
}  
  
int main() {  
 test();  
  
 size\_t n;  
 scanf("%zd", &n);  
 VectorVoid v = createVectorV(0, sizeof(float));  
 for (int i = 0; i < n; i++) {  
 float x;  
 scanf("%f", &x);  
 pushBackV(&v, &x);  
 }  
 for (int i = 0; i < n; i++) {  
 float x;  
 getVectorValueV(&v, i, &x);  
 printf("%f ", x);  
 }  
  
 return 0;  
}

**Результаты выполнения команд**

vlad@Mac-Pro-Vladislav programming-and-algorithmization-basics % git log --stat -- libs/data\_structures/vector/ main.c

commit 7a77565e39ef6f4c0b8c9c62c4ef8c8558a6e2ed (**HEAD**)

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 17:13:22 2023 +0300

refactoring

libs/data\_structures/vector/vector.c | 12 +++++++-----

libs/data\_structures/vector/vector.h | 10 +++++-----

main.c | 14 +++++++-------

3 files changed, 19 insertions(+), 17 deletions(-)

commit 5969e7fa04f8726163f210b69d048a2aad39a69d

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 17:02:32 2023 +0300

append access functions

libs/data\_structures/vector/vector.c | 15 +++++++++++++++

libs/data\_structures/vector/vector.h | 6 ++++++

main.c | 46 ++++++++++++++++++++++++++++++++++++++++++++++

3 files changed, 67 insertions(+)

commit 79781182e6be3a100b348a54e20daadfb10c940d

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 16:59:09 2023 +0300

append push / pop functions

libs/data\_structures/vector/vector.c | 28 ++++++++++++++++++++++++++++

libs/data\_structures/vector/vector.h | 12 ++++++++++++

main.c | 59 ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-

3 files changed, 98 insertions(+), 1 deletion(-)

commit df32e18a4b6750c08dfaddb55138bf27c654882c

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 16:54:51 2023 +0300

memory usage of vector

libs/data\_structures/vector/vector.c | 37 +++++++++++++++++++++++++++++++++++++

libs/data\_structures/vector/vector.h | 26 ++++++++++++++++++++++++++

main.c | 4 ++++

3 files changed, 67 insertions(+)

commit c8a082b9816f5cebe73ea6233a5694283d64a2f4

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 16:41:23 2023 +0300

first commit

main.c | 3 +++

1 file changed, 3 insertions(+)

vlad@Mac-Pro-Vladislav programming-and-algorithmization-basics % git log --stat -- libs/data\_structures/vectorVoid/

commit cc37a52e54212469a2b68fd0e5cacefe3dd6a6a8 (**HEAD -> c-lab14**, **origin/c-lab14**)

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 20:45:58 2023 +0300

refactoring / bug fix

libs/data\_structures/vectorVoid/vectorVoid.c | 4 ++++

1 file changed, 4 insertions(+)

commit 8c64de3f849674bea55200c29dab04f4781fda70

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 20:45:09 2023 +0300

append push / pop functions

libs/data\_structures/vectorVoid/vectorVoid.c | 34 ++++++++++++++++++++++++++++++++++

libs/data\_structures/vectorVoid/vectorVoid.h | 14 ++++++++++++++

2 files changed, 48 insertions(+)

commit 37ed17654cbdb9d57115fad9e30ad76ed31d37c2

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 20:43:47 2023 +0300

memory usage of vectorVoid

libs/data\_structures/vectorVoid/vectorVoid.c | 37 ++++++++++++++++++++++++++++++++++++-

libs/data\_structures/vectorVoid/vectorVoid.h | 15 +++++++++++++++

2 files changed, 51 insertions(+), 1 deletion(-)

commit 37aeb0602db97fb6111b5f6e626cd1a96ccf4cc6

Author: iamprogrammist <vladislav.pakhomov@bk.ru>

Date: Thu Apr 13 20:42:56 2023 +0300

init vectorVoid

libs/data\_structures/vectorVoid/vectorVoid.c | 1 +

libs/data\_structures/vectorVoid/vectorVoid.h | 14 ++++++++++++++

2 files changed, 15 insertions(+)

**Вывод:** в ходе выполнения лабораторной работы усовершенствованы навыки в создании библиотек, получение навыков работы с системой контроля версий 𝑔𝑖𝑡.