Московский Авиационный Институт

(Национальный исследовательский Университет)

Факультет: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**Лабораторная работа №5**

**по курсу «Операционные системы»**

|  |  |
| --- | --- |
| Студент: | Марков А.Н. |
| Группа: | М80-208Б-18 |
| Преподаватель: | Миронов Е.С. |
| Оценка: |  |
| Дата: |  |

Москва

2020

**Содержание**

1. Постановка задачи.
2. Общие сведение о программе.
3. Основные файлы программы.
4. Демонстрация работы программы.
5. Вывод.

**1. Постановка задачи.**

Требуется создать динамическую библиотеку, которая реализует определенный функционал. Далее использовать данную библиотеку 2-мя способами:

1. Во время компиляции (на этапе линковки).

2. Во время исполнения программы, подгрузив библиотеку в память с помощью системных вызовов.

В конечном счете, программа должна состоять из следующих частей:

1. Динамическая библиотека, реализующая заданный вариантом интерфейс.

2. Тестовая программа, которая использует библиотеку, используя знания полученные на этапе компиляции.

3. Тестовая программа, которая использует библиотеку, используя только местоположение динамической библиотеки и ее интерфейс.

**Вариант 3.**

**Структура данных, с которой должна обеспечивать работу библиотека:**

Работа с деком.

**Тип данных, используемый структурой.**

Целочисленный 32-битный.

**2. Общие сведения о программе.**

Была написана библиотека для дека. В его интерфейс входит создание дека, вставка в начало дека, вставка в конец дека, вставка по индексу, удаление из начала дека, удаление из конца дека, удаление по индексу, печать дека, возврат размера дека, очистка дека, возврат значения первого элемента дека, возврат значения последнего дека.

В случае линковки во время компиляции указываем путь до библиотеки в переменную среды LD\_LIBRARY\_PATH и указываем название библиотеки, пользуемся функциями как обычно.

В случае рантайм линковки нужно явно открыть библиотеку с помощью dlopen(), а затем присваивать указателям на функции результат dlsym(), который ищет по названию функции в библиотеке.

Системные вызовы, которые используются в программе:

void exit(int status) — программа завершается с заданным статусом.

void \*dlopen(const char \*filename, int flag) — открывает файл по пути filename со свойствами flag. В случае ошибки возвращается NULL. Flag обязательно должен иметь либо RTLD\_LAZE, либо RTLD\_NOW, которые отвечают за загрузку библиотеки (по частям, когда потребуется, либо все сразу).

void \*dlsym(void \*handle, const char \*symbol) — ищет в дереве подключенных через dlopen() библиотек строку symbol, если находит, то возвращает void \*участок памяти, связанный с функцией. В случае ошибки возвращается NULL.

int dlclose(void \*handle) — уменьшает количество ссылок на подключенную динамическую библиотеку, если он становится равным нулю, то библиотека отсоединяется. В случае успеха возвращает 0, иначе — не нуль.

**3. Основные файлы программы.**

**main1.c:**

#include <stdio.h>

#include <stdint.h>

#include "extern.h"

int menu() {

int choice;

printf("select\n"

"1-print\n"

"2-insert\n"

"3-delete\n"

"4-clear\n"

"5-push front\n"

"6-push back\n"

"7-pop front\n"

"8-pop back\n"

"9-size\n"

"10-front\n"

"11-back\n"

"12-end\n");

scanf("%d", &choice);

return choice;

}

int main() {

Deque deque;

deque\_init(&deque);

int choice, index;

int32\_t value;

while ((choice = menu()) != 12) {

switch (choice) {

case 1:

deque\_print(&deque);

break;

case 2:

printf("Enter index new value: ");

Deque\_iterator it;

it.ptr = deque.start;

scanf("%d", &index);

if (index < 0 || index > deque\_size(&deque)) {

printf("Error, invalid index\n");

break;

}

for (int j = 0; j < index; j++) {

it.ptr = it.ptr->next;

}

printf("Enter value: ");

int32\_t value;

scanf("%d", &value);

deque\_insert(it, value, &deque);

break;

case 3:

printf("Enter index to delete: ");

Deque\_iterator it1;

it1.ptr = deque.start;

scanf("%d", &index);

if (index < 0 || index >= deque\_size(&deque)) {

printf("Error, invalid index\n");

break;

}

for (int i = 0; i < index + 1; i++) {

it1.ptr = it1.ptr->next;

}

deque\_erase(it1, &deque);

break;

case 4:

deque\_clear(&deque);

break;

case 5:

printf("Enter value: ");

scanf("%d", &value);

deque\_push\_front(value, &deque);

break;

case 6:

printf("Enter value: ");

scanf("%d", &value);

deque\_push\_back(value, &deque);

break;

case 7:

deque\_pop\_front(&deque);

break;

case 8:

deque\_pop\_back(&deque);

break;

case 9:

printf("size: %d\n", deque\_size(&deque));

break;

case 10:

printf("front elem: %d\n", deque\_front(&deque));

break;

case 11:

printf("back elem: %d\n", deque\_back(&deque));

break;

default:

printf("Unknown command\n");

}

}

return 0;

}

**main2.c:**

#include <stdio.h>

#include <stdint.h>

#include "extern.h"

#include <dlfcn.h>

int menu() {

int choice;

printf("select\n"

"1-print\n"

"2-insert\n"

"3-delete\n"

"4-clear\n"

"5-push front\n"

"6-push back\n"

"7-pop front\n"

"8-pop back\n"

"9-size\n"

"10-front\n"

"11-back\n"

"12-end\n");

scanf("%d", &choice);

return choice;

}

int main() {

void \*library\_handler = dlopen("libdeque\_dyn.so", RTLD\_LAZY);

if (!library\_handler) {

printf("dlopen() error: %s\n", dlerror());

exit(1);

}

void (\*deque\_init)(Deque \*d);

int (\*deque\_size)(Deque \*d);

void (\*deque\_clear)(Deque \*d);

void (\*deque\_push\_front)(int32\_t num, Deque \*d);

void (\*deque\_push\_back)(int32\_t num, Deque \*d);

void (\*deque\_insert)(Deque\_iterator it, int32\_t num, Deque \*d);

void (\*deque\_pop\_front)(Deque \*d);

void (\*deque\_pop\_back)(Deque \*d);

void (\*deque\_erase)(Deque\_iterator it, Deque \*d);

int32\_t (\*deque\_front)(Deque \*d);

int32\_t (\*deque\_back)(Deque \*d);

void (\*deque\_print)(Deque \*d);

deque\_init = dlsym(library\_handler, "deque\_init");

deque\_size = dlsym(library\_handler, "deque\_size");

deque\_clear = dlsym(library\_handler, "deque\_clear");

deque\_push\_front = dlsym(library\_handler, "deque\_push\_front");

deque\_push\_back = dlsym(library\_handler, "deque\_push\_back");

deque\_insert = dlsym(library\_handler, "deque\_insert");

deque\_pop\_front = dlsym(library\_handler, "deque\_pop\_front");

deque\_pop\_back = dlsym(library\_handler, "deque\_pop\_back");

deque\_erase = dlsym(library\_handler, "deque\_erase");

deque\_front = dlsym(library\_handler, "deque\_front");

deque\_back = dlsym(library\_handler, "deque\_back");

deque\_print = dlsym(library\_handler, "deque\_print");

Deque deque;

deque\_init(&deque);

int choice, index;

int32\_t value;

while ((choice = menu()) != 12) {

switch (choice) {

case 1:

deque\_print(&deque);

break;

case 2:

printf("Enter index new value: ");

Deque\_iterator it;

it.ptr = deque.start;

scanf("%d", &index);

if (index < 0 || index > deque\_size(&deque)) {

printf("Error, invalid index\n");

break;

}

for (int j = 0; j < index; j++) {

it.ptr = it.ptr->next;

}

printf("Enter value: ");

int32\_t value;

scanf("%d", &value);

deque\_insert(it, value, &deque);

break;

case 3:

printf("Enter index to delete: ");

Deque\_iterator it1;

it1.ptr = deque.start;

scanf("%d", &index);

if (index < 0 || index >= deque\_size(&deque)) {

printf("Error, invalid index\n");

break;

}

for (int i = 0; i < index + 1; i++) {

it1.ptr = it1.ptr->next;

}

deque\_erase(it1, &deque);

break;

case 4:

deque\_clear(&deque);

break;

case 5:

printf("Enter value: ");

scanf("%d", &value);

deque\_push\_front(value, &deque);

break;

case 6:

printf("Enter value: ");

scanf("%d", &value);

deque\_push\_back(value, &deque);

break;

case 7:

deque\_pop\_front(&deque);

break;

case 8:

deque\_pop\_back(&deque);

break;

case 9:

printf("size: %d\n", deque\_size(&deque));

break;

case 10:

printf("front elem: %d\n", deque\_front(&deque));

break;

case 11:

printf("back elem: %d\n", deque\_back(&deque));

break;

default:

printf("Unknown command\n");

}

}

int res\_close = dlclose(library\_handler);

if (res\_close != 0) {

printf("dlclose() error\n");

}

return 0;

}

**extern.h:**

#ifndef EXTERN\_H

#define EXTERN\_H 1

#include <stdint.h>

#include <stdlib.h>

#include <stdio.h>

typedef struct Deque\_item Deque\_item;

typedef struct Deque Deque;

typedef struct Deque\_iterator Deque\_iterator;

struct Deque\_item {

int32\_t value;

Deque\_item \*next;

Deque\_item \*prev;

};

struct Deque {

Deque\_item \*start;

Deque\_item \*end;

int size;

};

struct Deque\_iterator {

Deque\_item \*ptr;

};

extern void deque\_init(Deque \*);

extern int deque\_size(Deque \*);

extern void deque\_clear(Deque \*);

extern void deque\_push\_front(int32\_t, Deque \*);

extern void deque\_push\_back(int32\_t, Deque \*);

extern void deque\_insert(Deque\_iterator, int32\_t, Deque \*);

extern void deque\_pop\_front(Deque \*);

extern void deque\_pop\_back(Deque \*);

extern void deque\_erase(Deque\_iterator, Deque \*);

extern int32\_t deque\_front(Deque \*);

extern int32\_t deque\_back(Deque \*);

extern void deque\_print(Deque \*);

#endif //EXTERN\_H

**deque.c:**

#include <stdint.h>

#include <stdlib.h>

#include <stdio.h>

#include "extern.h"

void deque\_init(Deque \*d) {

d->start = (Deque\_item \*) malloc(sizeof(Deque\_item));

d->end = (Deque\_item \*) malloc(sizeof(Deque\_item));

d->start->next = d->end;

d->start->prev = d->end;

d->end->next = d->start;

d->end->prev = d->start;

d->size = 0;

}

int deque\_size(Deque \*d) {

return d->size;

}

void deque\_clear(Deque \*d) {

if (d->size == 0) {

printf("Deque is empty.\n");

return;

}

Deque\_item \*iter = d->start->next;

int sz\_before\_dstr = deque\_size(d);

for (int i = 0; i < sz\_before\_dstr; i++) {

Deque\_item \*m = iter;

iter->prev->next = iter->next;

iter->next->prev = iter->prev;

iter = iter->next;

free(m);

d->size--;

}

}

void deque\_push\_front(int32\_t num, Deque \*d) {

d->size++;

Deque\_item \*temp = (Deque\_item \*) malloc(sizeof(Deque\_item));

temp->value = num;

temp->next = d->start->next;

d->start->next->prev = temp;

temp->prev = d->start;

d->start->next = temp;

}

void deque\_push\_back(int32\_t num, Deque \*d) {

d->size++;

Deque\_item \*temp = (Deque\_item \*) malloc(sizeof(Deque\_item));

temp->value = num;

d->end->prev->next = temp;

temp->prev = d->end->prev;

temp->next = d->end;

d->end->prev = temp;

}

void deque\_insert(Deque\_iterator it, int32\_t num, Deque \*d) {

d->size++;

Deque\_item \*temp = (Deque\_item \*) malloc(sizeof(Deque\_item));

temp->value = num;

temp->next = it.ptr->next;

it.ptr->next->prev = temp;

temp->prev = it.ptr;

it.ptr->next = temp;

}

void deque\_pop\_front(Deque \*d) {

d->size--;

Deque\_item \*temp = d->start->next;

d->start->next = temp->next;

temp->next->prev = temp->prev;

free(temp);

}

void deque\_pop\_back(Deque \*d) {

d->size--;

Deque\_item \*temp = d->end->prev;

d->end->prev = temp->prev;

temp->prev->next = temp->next;

free(temp);

}

void deque\_erase(Deque\_iterator it, Deque \*d) {

d->size--;

Deque\_item \*temp = it.ptr;

it.ptr->prev->next = it.ptr->next;

it.ptr->next->prev = it.ptr->prev;

free(temp);

}

int32\_t deque\_front(Deque \*d) {

if (d->size == 0) {

printf("Deque is empty.\n");

return 0;

}

return d->start->next->value;

}

int32\_t deque\_back(Deque \*d) {

if (d->size == 0) {

printf("Deque is empty.\n");

return 0;

}

return d->end->prev->value;

}

void deque\_print(Deque \*d) {

if (d->size == 0) {

printf("Deque is empty.\n");

return;

}

Deque\_item \*temp = d->start->next;

for (int i = 0; i < d->size; i++) {

printf("%d ", temp->value);

temp = temp->next;

}

printf("\n");

}

**Makefile:**

all: dyn2 dyn1

dyn2: main2.c

gcc -o dyn2 main2.c -ldl -g

dyn1: main1.c deque.c

gcc -fPIC -c deque.c

gcc -shared -o libdeque\_dyn.so deque.o

gcc -o dyn1 main1.c -L . -l deque\_dyn

**4. Демонстрация работы программы.**

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ pwd

/home/oem/Documents/OS/OS/lab05/src

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ LD\_LIBRARY\_PATH=/home/oem/Documents/OS/OS/lab05/src

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ export LD\_LIBRARY\_PATH

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ make

gcc -o dyn2 main2.c -ldl

gcc -fPIC -c deque.c

gcc -shared -o libdeque\_dyn.so deque.o

gcc -o dyn1 main1.c -L . -l deque\_dyn

**1. Использование динамической библиотеки, присоединенной к программе во время линковки.**

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ ./dyn1

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

2

Enter index new value: 0

Enter value: 10

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

5

Enter value: 5

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

6

Enter value: 15

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

5 10 15

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

3

Enter index to delete: 1

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

5 15

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

9

size: 2

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

10

front elem: 5

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

11

back elem: 15

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

7

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

8

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

Deque is empty.

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

2

Enter index new value: 0

Enter value: 5

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

2

Enter index new value: 1

Enter value: 10

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

2

Enter index new value: 15

Error, invalid index

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

5 10

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

4

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

Deque is empty.

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

12

**2. Использование динамической библиотеки, подключая ее во время работы программы.**

oem@Alex-PC:~/Documents/OS/OS/lab05/src$ ./dyn2

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

5

Enter value: 10

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

6

Enter value: 15

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

6

Enter value: 20

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

1

10 15 20

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

10

front elem: 10

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

11

back elem: 20

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

9

size: 3

select

1-print

2-insert

3-delete

4-clear

5-push front

6-push back

7-pop front

8-pop back

9-size

10-front

11-back

12-end

12

**6. Вывод.**

Использование динамических библиотек в проектах позволяет уменьшить размер исполняемого файла, а также при использовании одной и той же библиотеки несколькими программами, не придется загружать ее в память несколько раз.

Работа с динамическими библиотеками может быть полезна при низкоуровневой реализации паттерна plug-in, который заключается в том, что можно подключать и отключать библиотеки во время работы.