Отчет по лабораторной работе №25-26 по курсу ЯМП

Студент группы М8О-107Б-19 Носов Эмиль Лаймонасович, № по списку 17

Контакты www, e-mail, icq, skype_____

Работа выполнена: «27» марта 2021г.

Преподаватель: доцент каф. 806 Сластушенский Ю. В.

Входной контроль знаний с оценкой _____

Отчет сдан « » _____201 __ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Автоматизация сборки программ модульной структуры на языке Си с использованием утилиты make, абстрактные типы данных, рекуурсия, модульные программирование на языке Си.

2. **Цель работы:** Изучить принцип работы утилиты make, составить Makefile для модульной программы из лабораторной работы №26, протестировать makefile в различных ситуациях, составить и отладить модули определений и реализации по заданной схеме, составить программный модуль , сортирующий экземпляр указанного абстрактного типа данных заданным методом.
3. **Задание** (*вариант № 17*)**:** АТД — дек; процедура и метод — поиск и удаление максимального элемента, сортировка линейным выбором.
4. **Оборудование** (лабораторное):
   ЭВМ _____, процессор _____, имя узла сети _____ с ОП _____ Мб,
   НМД _____ Мб. Терминал _____ адрес _____. Принтер _____
   Другие устройства _____
   _____

   Оборудование ПЭВМ студента, если использовалось:
   Процессор: Ryzen 5 4600H с ОП 16 Гб, НМД 256 Гб. Монитор 1920x1080
   Другие устройства _____
   _____

5. **Программное обеспечение (лабораторное):**
   Операционная система семейства _____, наименование _____ версия _____
   интерпретатор команд _____ версия _____
   Система программирования _____ версия _____
   Редактор текстов _____ версия _____
   Утилиты операционной системы _____
   _____
   Прикладные системы и программы _____
   Местонахождение и имена файлов программ и данных_____
   _____

   Программное обеспечение ЭВМ студента, если использовалось:
   Операционная система семейства Windows, наименование Windows 10 версия 2004
   интерпретатор команд bash версия 5.0.17.
   Система программирования _____версия _____
   Редактор текстов _____ версия _____
   Утилиты операционной системы _____
   _____
   Прикладные системы и программы _____
   _____
   Местонахождение и имена файлов программ и данных на домашнем компьютере _____
   _____

6. **Идея, метод, алгоритм** решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)

   При сортировке линейным выбором из дека удаляется самый большой элемент и вставляется в начало другого дека, так повторяется до тех пор, пока сортируемый дек не опустеет. После этого он заменяется на дек, в начало которого каждый раз помещали максимальные значения, который является отсортированным.

7. **Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

Makelab25

```
#makefile lab25
CC = gcc
LD = gcc
CCFLAGS = -Wall -pedantic -std=c11
LDFLAGS =
lab26: lab26.o deq.o sort.o
	$(LD) $(LDFLAGS) -o lab26 lab26.o deq.o sort.o
lab26.o: lab26.c deq.h
	$(CC) $(CCFLAGS) -c lab26.c
deq.o: deq.c deq.h
	$(CC) $(CCFLAGS) -c deq.c
sort.o: sort.c deq.h deq.c
	$(CC) $(CCFLAGS) -c sort.c
clean:
	rm -f *.o
```

deq.h

```
#ifndef DEQ_H
#define DEQ_H

#pragma once

#include<stdio.h>
#include<stdlib.h>

typedef struct deque deque;
typedef struct deque *Deque;

struct deque{
	int *key;
	int size;
};

Deque NewDeq();

void print(Deque);

int Isempty(Deque);

void PushBack(Deque, int);

void PushFront(Deque, int);

int PopBack(Deque);

int PopFront(Deque);

void Clear(Deque);
```

```
int DeleteMax(Deque);

void Sort(Deque);

int IsEmpty(Deque);

#endif

#include "deq.h"
```

deq.c

```
Deque NewDeq(){
	Deque a=(Deque)malloc(sizeof(deque));
	a->size=0;
	a->key=NULL;
	return(a);
}

void print(Deque a){
	for(int i=0; i<a->size; i++){
		printf("%d ",a->key[i]);
	}
	printf("\n");
}

int IsEmpty(Deque a){
	int b=0;
	if(a->size==0) b=1;
	return(b);
}

void PushBack(Deque a, int b){
	if(a->key==NULL){
		a->key=(int*)malloc(sizeof(int));
		a->key[0]=b;
	}else{
		int *c=(int*)malloc((a->size+1)*sizeof(int));
		c[a->size]=b;
		for(int i=0; i<a->size; i++){
			c[i]=a->key[i];
		}
		free(a->key);
		a->key=c;
	}
	a->size++;
}
```

```
void PushFront(Deque a, int b){
	if(a->key==NULL){
		a->key=(int*)malloc(sizeof(int));
		a->key[0]=b;
	}else{
		int *c=(int*)malloc((a->size+1)*sizeof(int));
		c[0]=b;
		for(int i=1; i<=a->size; i++){
			c[i]=a->key[i-1];
		}
		free(a->key);
		a->key=c;
	}
	a->size++;
}

int PopBack(Deque a){
	int b=0;
	if(a->size>0){
		b=a->key[a->size-1];
		int*c=(int*)malloc((a->size-1)*sizeof(int));
		for(int i=0; i<a->size-1; i++){
			c[i]=a->key[i];
		}
		free(a->key);
		a->key=c;
		a->size--;
	}
	if (a->size==0){
		a->size=0;
		a->key=NULL;
	}
	return(b);
}

int PopFront(Deque a){
	int b=0;
	if(a->size>0){
		b=a->key[0];
		int*c=(int*)malloc((a->size-1)*sizeof(int));
		for(int i=0; i<a->size-1; i++){
			c[i]=a->key[i+1];
		}
		free(a->key);
		a->key=c;
		a->size--;
```

```
	}
	if (a->size==0){
		a->size=0;
		a->key=NULL;
	}
	return(b);
}

void Clear(Deque a){
	a->size=0;
	free(a->key);
	a->key=NULL;
}
```

sort.c

```
#include "deq.h"

int DeleteMax(Deque a){
	int b=a->size;
	int c=0, max=0;
	for(int i=0; i<b; i++){
		c=PopBack(a);
		if(c>max) max=c;
		PushFront(a, c);
	}
	for(int i=0; i<b; i++){
		c=PopBack(a);
		if(c!=max) PushFront(a, c); else break;
	}
	return(max);
}

void Sort(Deque a){
	int c, b=a->size;
	Deque deq=NewDeq();
	for(int i=0; i<b; i++){
		c=DeleteMax(a);
		PushFront(deq, c);
	}
	*a=*deq;
}
```

lab26.c

```
#include "deq.h"

int main(){
	Deque deq=NewDeq();
```

```
	char a;
	int b;
	printf("\n B - PushBack \n F - PushFront \n b - PopBack \n f - PopFront \n s - Sort \n p - Print \n i - Is empty \n c - Clear \n q - quit \n\n");
	while(1>0){
	if(a!='\n')printf("Enter command: ");
	scanf("%c",&a);
	if(a!='\n')printf("\n");
	switch(a){
		case 'B':
				printf("Enter value: ");
				scanf("%d",&b);
				PushBack(deq, b);
				printf("\n");
				break;
		case 'F':
				printf("Enter value: ");
				scanf("%d",&b);
				PushFront(deq, b);
				printf("\n");
				break;
		case 'b':
				if(deq->key!=NULL){
				printf("%d",PopBack(deq));
				}else printf("\nIt is empty");
				printf("\n");
				break;
		case 'f':
				if(deq->key!=NULL){
				printf("%d",PopFront(deq));
				}else printf("\nIt is empty");
				printf("\n");
				break;
		case 's':
				print(deq);
				Sort(deq);
				printf("\n");
				print(deq);
				printf("\n");
				break;
		case 'p':
				print(deq);
				printf("\n");
				break;
		case 'c':
				Clear(deq);
				printf("\nDequeue size is %d\n",deq->size);
```

```
                    break;
            case 'i':
                    printf("\n");
                    if(IsEmpty(deq)) printf("Deque is empty");
                    else printf("Deque isn't empty");
                    printf("\n");
                    break;
            case 'q': return (0);
        }
        }
}
```

Пункты 1-7 отчета составляются сторого до начала лабораторной работы.

Допущен к выполнению работы. **Подпись преподавателя** _____

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

```
$ make -f makelab25
gcc -Wall -pedantic -std=c11 -c lab26.c
gcc -Wall -pedantic -std=c11 -c deq.c
gcc -Wall -pedantic -std=c11 -c sort.c
gcc  -o lab26 lab26.o deq.o sort.o
$ ls
deq.c  deq.h  deq.h.gch  deq.o  lab26  lab26.c  lab26.o  makelab25  sort.c  sort.o  лр26.exe  лр26.h.gch
$ touch deq.h
$ make -f makelab25
gcc -Wall -pedantic -std=c11 -c lab26.c
gcc -Wall -pedantic -std=c11 -c deq.c
gcc -Wall -pedantic -std=c11 -c sort.c
gcc  -o lab26 lab26.o deq.o sort.o
$ touch lab26.c
$ make -f makelab25
gcc -Wall -pedantic -std=c11 -c lab26.c
gcc  -o lab26 lab26.o deq.o sort.o
$ make -f makelab25
make: 'lab26' is up to date.
$./lab26

 B - PushBack
 F - PushFront
 b - PopBack
 f - PopFront
 s - Sort
 p - Print
 i - Is empty
 c - Clear
 q - quit

Enter command: B
```

```
Enter value: 6

Enter command: F

Enter value: 9

Enter command: B

Enter value: 8

Enter command: F

Enter value: 6

Enter command: p

6 9 6 8

Enter command:
b

8
Enter command: f

6
Enter command: s

9 6

6 9

Enter command: i


Deque isn't empty
Enter command: c


Dequeue size is 0
Enter command: p



Enter command: q
```

9. **Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание

10. **Замечания автора** по существу работы _____

_____

_____

_____

11. **Выводы**

В ходе работы я изучил принцип работы утилиты make, составить Makefile для модульной программы из лабораторной работы №26, протестировал makefile в различных ситуациях, составил и отладил модули определений и реализации абстактного типа данных дек, составить программный модуль , сортирующий экземпляр дека методом линейного выбора с использованием функции поиска и удаления максимального элемента.

Недочёты при выполнении задания могут быть устранены следующим образом: _____

_____

_____

_____

_____

_____

Подпись студента _____