Основы алгоритмизации и программирования

ФИСТ 1 курс
Власенко
Олег
Федосович

Лекция 15.1

Экзамен:

Структура. Требования. Задания.

Минимальный (обязательный) набор практических навыков

1. Умение нарисовать блок-схему алгоритма для линейных участков, любых циклов, развилок, в том числе вложенных (любой степени вложенности).
2. Умение выполнять ручную трассировка алгоритма/программы – на листочке бумаги.
3. Умение создать новый проект в VS. Умение набрать текст программы в редакторе VS.
4. Умение собрать и запустить программу на выполнение в VS.
5. Умение ввести числовые данные с клавиатуры в консоли. Умение прочитать результат программы в консоли.
6. Умение трассировать программу в VS – линейные участки, развилки, циклы. Умение трассировать вызов функций, трассировать рекурсию. Просматривать локальные переменные, стек вызовов, любые выбранные выражения.
7. Умение создавать текстовые файлы вручную.
8. Умение читать программно числовые данные из текстовых файлов.
9. Умение создавать проект Windows.
10. Умение создавать картинки средствами Win API, состоящие из линий, прямоугольников и эллипсов.

Минимальный набор алгоритмов (выученных наизусть!)

1. Вычисление факториала.
2. Вычисление чисел Фибоначчи.
3. Ввод элементов массива
4. Вывод элементов массива
5. Поиск минимального/максимального элемента массива
6. Поиск среднего арифметического элементов массива
7. Обмен местами двух элементов массива
8. Функция strlen - смочь написать свою реализацию
9. Функция strcat - смочь написать свою реализацию
10. Функция strcpy - смочь написать свою реализацию

Пример экзаменационной задачи (на «хорошо»)

Общее задание

- Ввести двумерный массив из файла. Количество элементов не более 10x10.
- Каждый элемент – целое число в интервале значений -1000..+1000.
- Количество строк (N) и столбцов (M) задано первой строке входного файла. Далее в N строках записаны по M числе.
- Обработать массив согласно варианту.
- Сохранить результат в формате, аналогичном входному

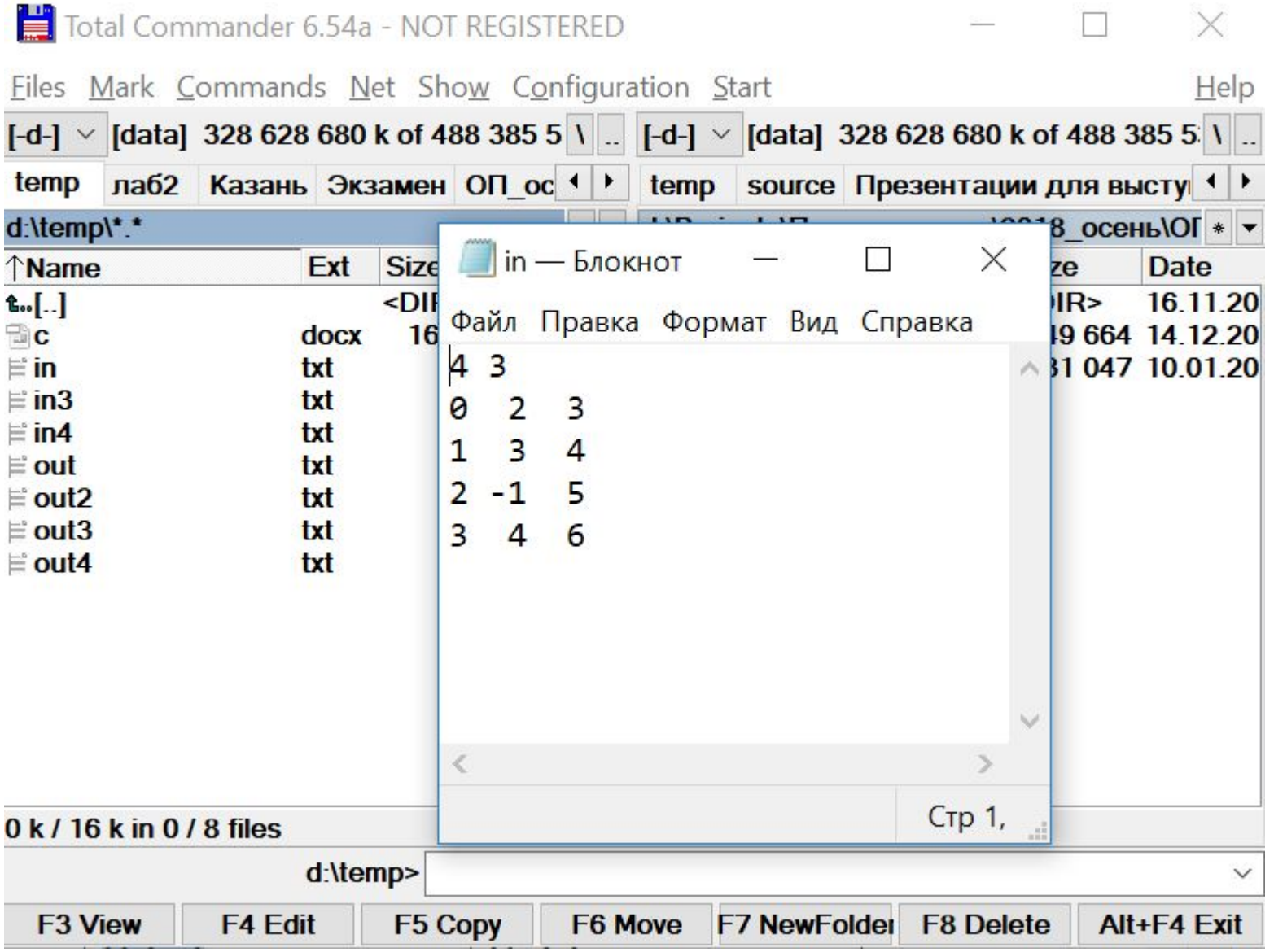
Вариант

Номер варианта	Задание	Входные данные (вводятся из текстового файла)	Выходные данные (выводятся в текстовый файл)
B0	Переставить столбцы, содержащие минимальный и максимальный элементы	4 3 0 2 3 1 3 4 2 -1 5 3 4 6	4 3 0 3 2 1 4 3 2 5 -1 3 6 4

Простое решение экзаменационной задачи

(другие варианты решения – см. Лекция 9)

Входной файл



Решение без функций (1)

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

void main()
{
	int a[10][10];
	int n, m;

	// Чтение из входного файла
	FILE *fin = fopen("d:\\Temp\\in.txt", "rt");
	if (fin == NULL) {
		printf("File in3.txt is not found");
		return;
	}

	fscanf(fin, "%d", &n);
	fscanf(fin, "%d", &m);
	for (int i = 0; i < n; i++)
		for (int j = 0; j < m; j++)
			fscanf(fin, "%d", &a[i][j]);
	fclose(fin);
```

Решение без функций (2)

```
// вывод массива в консоль
for (int i = 0; i < n; i++) {
	for (int j = 0; j < m; j++) {
		printf("%5d ", a[i][j]);
	}
	printf("\n");
}
printf("\n");

// поиск минимального
int min = a[0][0];
int iMin = 0;
int jMin = 0;
for (int i = 0; i < n; i++) {
	for (int j = 0; j < m; j++) {
		if (a[i][j] < min) {
			min = a[i][j];
			iMin = i;
			jMin = j;
		}
	}
}
```

Решение без функций (3)

```
// поиск максимального
int max = a[0][0];
int iMax = 0;
int jMax = 0;
for (int i = 0; i < n; i++) {
	for (int j = 0; j < m; j++) {
		if (a[i][j] > max) {
			max = a[i][j];
			iMax = i;
			jMax = j;
		}
	}
}

// перестановка столбцов с минимальным и максимальным элементами
for (int i = 0; i < n; i++) {
	int tmp = a[i][jMin];
	a[i][jMin] = a[i][jMax];
	a[i][jMax] = tmp;
}
```

Решение без функций (4)

```
// вывод массива в консоль
for (int i = 0; i < n; i++) {
	for (int j = 0; j < m; j++) {
		printf("%5d ", a[i][j]);
	}
	printf("\n");
}
printf("\n");

// Запись в выходной файл
FILE *fout = fopen("d:\\Temp\\out.txt", "wt");
if (fout == NULL) {
	printf("File out.txt cannot be created");
	return;
}

for (int i = 0; i < n; i++) {
	for (int j = 0; j < m; j++)
		fprintf(fout, "%5d ", a[i][j]);
	fprintf(fout, "\n");
}
fclose(fout);
```

Решение без функций (5)

```
    {
        int x;
        scanf("%d", &x);
    }
}
```

Пример экзаменационной задачи (на «отлично»)

Общее задание

- Ввести двумерный массив из файла. Количество элементов не более 10x10.
- Каждый элемент – целое число в интервале значений -1000..+1000.
- Количество строк (N) и столбцов (M) задано первой строке входного файла. Далее в N строках записаны по M числе.
- Обработать массив согласно варианту.
- Сохранить результат в формате, аналогичном входному.
- Имейте в виду - во время выполнения программы количество используемых строк и столбцов может изменится. Предусмотрите в массиве столько строк и столбцов, чтобы программа корректно работала на любых корректных входных данных.
- Нельзя использовать дополнительные массивы – в программе используется ровно один массив.

Пример экзаменационной задачи (на «отлично»)

Вариант

Номер варианта	Задание	Входные данные (вводятся из текстового файла)	Выходные данные (выводятся в текстовый файл)
A0	Удалить все строки, содержащие отрицательные числа	4 3 0 2 3 1 3 -4 2 -1 5 3 4 5	2 3 0 2 3 3 4 5

Теоретические вопросы к экзамену

1. Графические схемы алгоритмов = блок-схемы алгоритмов. (Линейный алгоритм, развилка, цикл, вложенный цикл и т.д., break, continue, вызов функции, в том числе рекурсивной, return)
2. Идентификаторы (DlinnoeMnemonicheskoeImya1).
3. Стандартные типы данных (int).
4. **sizeof**(), диапазон значений типа (Пр: unsigned char – от 0 до 255).
5. Подключение библиотек (**#include**)
6. Объявление переменных (int a;).
7. Выражения (a+b*c).
8. Операторы. Приоритеты операторов (-b+sqrt(d)/2*a).
9. Развилка. Полная, усеченная, вложенная. (**if** (a<b) min = a;)
10. Выбор (**switch**(day) { **case** Monday: printf("Mn");}).
11. Циклы. Циклы с предусловием и с постусловием (**while** (*str++);).
12. Циклы для обхода всех элементов. Цикл for (**for**(i=0; i<n;++i) …).
13. Изменение естественного хода выполнения программы – инструкции **break**, **continue**, **return**, goto и т.п.)
14. Структуры (**struct** Line {int x1, y1, x2, y2;}; struct Line *p; …p->x1 = 10;).
15. Указатели (**int ***p; p = &a; *p = 10;). Указатель **void ***
16. Функции. (int f(int x) {}).
17. Область видимости переменных. Локальные и глобальные переменные (int x; void f() { int x;}).
18. Статические переменные (void f() { **static** int x = 0;}).

Теоретические вопросы к экзамену

19. Разделы памяти во время выполнения программы: статическая, автоматическая, динамическая, машинный код.
20. Рекурсия. Прямая и косвенная. Область применения. (void f() { f();})
21. Массивы. Работа с одномерными массивами. Работа с двумерными массивами. (int a[10];)
22. Динамическая память. Выделение и освобождение динамической памяти. (malloc, free).
23. Символы. ASCII. Функции обработки символов (if (isdigit(ch) {digit = ch – '0';})
24. Обработка текста. Строки ASCIIZ. (char s[]= "abc"; int len = strlen(s);)
25. Ввод/вывод. Консоль. (scanf("%d", &a); printf("%d", a * a);)
26. Ввод/вывод. Текстовые файлы. HTML. (FILE * f = fopen(filename, "rt");)
27. Создание многомодульных проектов.
28. Декартова система координат. Экранная система координат. Рисование линий средствами Win API.
29*. Реализация односвязных списков. (struct Item {Data data; Item * next;};)
30*. Реализация двусвязных списков. (struct Item {Data data; Item * next; Item * prev;};)
31*. Понятие «Двоичное дерево поиска». Реализация двоичного дерева поиска на Си. (struct NodeTree { int data; NodeTree * left; NodeTree * right; };)
32*. Вычислительная сложность алгоритма. Асимптотическая оценка сложности: O(1), O(log N), O(N), $O(N^2)$, … . Измерение времени работы программы.

Вопросы помеченные "" (с 29 по 32) являются обязательными для сдачи на оценку «отлично». Они исключены из списка вопросов на «хорошо» и

Ориентировочная структура экзамена

1. Сделать все лаб работы
2. Сдать все лаб работы
3. Подготовиться к экзамену
   а) Выполнить пробные задачи
   б) выбрать уровень сложности, который вы готовы подтверждать на экзамене
   в) Подготовить ответы на теоретические вопросы
4. * Посетить консультацию перед экзаменом – задать на ней любые вопросы по предмету и списку вопросов, получить ответы
5. Прийти на экзамен
6. Выбрать уровень сложности
7. Получить практическую задачу соответствующего уровня сложности
8. За ограниченное время - выполнить полученную задачу
9. Продемонстрировать работоспособность задачи на любых корректных входных данных
10. В случае необходимости – отладить и исправить код
11. По требованию преподавателя – нарисовать блок-схему для указанного фрагмента кода
12. По требованию преподавателя – продемонстрировать навыки трассировки как вручную так и в VS
13. Выбрать теоретические вопросы
14. Подготовить ответы письменно
15. В беседе с преподавателем показать знакомство с темами теоретических вопросов
16. В случае необходимости – ответить на дополнительные вопросы без подготовки
17. Получить оценку
18. Отметить успешную сдачу экзамена

Уровни сложности на экзамене

На «удовлетворительно»

1. Можно сдать не все лабораторные работы (Пибд и ИСЭбд – 12 и 14 лабы можно не сдавать!)
2. Задача практическая (из списка Обязательных алгоритмов)
3. 1 теоретический вопрос – при подготовке можно пользоваться бумажными носителями информации
4. Никаких дополнительных вопросов

На «хорошо»

1. Сделаны и сданы все лабораторные работы
2. Задача практическая
3. 2 теоретических вопроса – при подготовке можно пользоваться бумажными носителями информации
4. дополнительные вопросы – 1-2

На «отлично»

1. Сделаны и сданы все лабораторные работы.
2. Оценка за аттестацию – не ниже «4» ИЛИ есть особые дополнительные заслуги (победитель олимпиады в течение семестра и т.п.)
3. Задача практическая
4. 2 теоретических вопроса – при подготовке нельзя пользоваться ничем
5. дополнительные вопросы – 6+

Рекомендуемая литература

1. Конспект лекций
2. Презентации к лекциям
3. Ссылки в презентациях, в том числе следующие:

- http://c-spravochnik.ru/
- https://msdn.microsoft.com/ru-ru/default.aspx
- http://habrahabr.ru/
- https://www.google.ru/
- http://rsdn.ru/