# ­Практические работы на 2 семестр

Напоминаю о том, что в этом семестре у вас ЭКЗАМЕН, при выставлении оценки за который необходимо учитывать ваши достижения в семестре, вследствие чего все работы этого семестра будут оцениваться. Помимо самой оценки за работу, напоминаю о важном принципе – если работа сдана после срока – оценка за неё сразу же снижается вдвое. То есть если вы принесёте первую работу после 25 марта – в лучшем случае полУчите за неё 4 балла вместо 8. Также рекомендую вам перечитать описание к работам прошлого семестра и требования к процессу их сдачи. В прошлом семестре я был добр, но в этом такого не будет. Кратко напомню о чём там была речь:

1. Программа должна выполняться только самостоятельно.

2. Перед выполнением любой работы, прежде всего, необходимо разобраться с теоретическими аспектами её выполнения, причём по возможности самостоятельно.

3. В течение одного занятия допускается сдача максимум одной лабораторной работы.

4. Подходите к написанию программ и оценке качества выполненной вами работы ответственно и осмысленно.

График сдачи работ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер | Название | Время выполнения | Крайний срок сдачи | Максимальный балл |
| 1 | Работа с файлами /  обработка строки | 2 недели | 25 марта | 8 |
| 2 | Работа с динамическим массивом | 1 неделя | 1 апреля | 8 |
| 3 | Работа с циклическим списком | 2 недели | 15 апреля | 10 |
| 4 | Проверка возможности хода | 1 неделя | 22 апреля | 6 |
| 5 | Отображение | 1 неделя | 29 апреля | 6 |
| 6 | Архипелаг /  Экваториальный остров | 2 недели | 13 мая | 20 |
| 7 | Команда MS-DOS | 2 недели | 27 мая | 12 |

## Работа №1 (вариант 1): Работа с файлами

Написать программу, которая выполнит следующие действия:

1. сформирует файл, который содержит некоторое количество случайных целых чисел некоторой длины. Длины всех чисел одинаковые, каждое число не менее чем двузначное (можете взять любое число разрядов какое нравится и даже вынести его в define-константу).

2. прочитать числа из сформированного файла в случайном порядке и записать их в другой файл таким образом, чтобы никакое из чисел не присутствовало в результирующем файле дважды (кроме случаев, когда оно дважды или более встречается в исходном файле).

Программа должна обрабатывать аргументы командной строки. Первый аргумент – имя формируемого файла, второй аргумент – имя файла с результатом (если этот аргумент отсутствует, то результат необходимо записать в файл с именем result.txt). Пример вызова:

generatenumbers.exe source.txt target.txt

В случае если содержимое первого файла сгенерируется, например, таким (тут оно не сильно похоже на случайное, но зато так будет лучше понять смысл, я надеюсь):

11 22 33 44 55 66 77

Тогда файл target.txt сможет получить, например, такое содержимое:

33 11 77 22 66 55 44

То есть все числа из исходного файла расположены в новом файле в случайном порядке.

Время на выполнение – 2 недели

## Работа №1 (вариант 2): Обработка строки

Из файла, указанного в командной строке при запуске программы, необходимо считать выражение, содержащие скобки четырёх видов – круглые (), квадратные [], фигурные {} и угловые <>. Необходимо проверить, является ли последовательность скобок корректной, а именно:

1. Что любая открывающая скобка имеет закрывающую того же типа где-то дальше по строке

2. Что пары скобок не пересекаются (вложенные скобки при этом допустимы):

Пример выражения с правильно расставленными скобками:

(a+[b\*c] <- {d/3}>)

Пример выражения с неправильно расставленными скобками:

(a+[b\*c) - 17]

Ответ необходимо вывести на экран в удобочитаемом виде. Из ответа должно быть понятно, верно ли расставлены скобки каждого вида и если неверно – то указать какие именно скобки расставлены неверно.

Например, для вот такого выражения:

4 (a + [b \* c] - {d / 3}) a b {c} d>

результат, вероятно, будет следующим:

() - верно

[] - верно

{} - верно

<> - неверно

Время на выполнение – 2 неделя

## Работа №2: Работа с динамическим массивом

Необходимо написать программу, которая позволяет работать с динамическим массивом ВЕЩЕСТВЕННЫХ чисел с помощью меню, в котором есть следующие пункты:

1. Задать размер массива

2. Заполнить массив

3. Вывести содержимое массива на экран

4. Выйти

Операция 1 позволяет выделить новую память для массива. При повторном вызове данного пункта меню, в случае если массив уже был заполнен, заполненные значения необходимо сохранить. Если элементов в массиве стало больше чем было, то добавленные элементы необходимо заполнить нулями. Выделение памяти под массив нового размера необходимо оформить в виде отдельной функции.

Операция 2 позволяет заполнить массив значениями. Значения необходимо спросить у пользователя. Если при выборе пункта 2 размер массива ещё не задан – необходимо выполнить всё то, что требуется при выборе операции 1.

Операции 3 и 4, как я полагаю и как я надеюсь - очевидны.

Обратите внимание, что поскольку число элементов массива изначально неизвестно, то объявить его придётся в виде указателя, например, вот так:

double \*a;

Для того чтобы работать с этим массивом вам потребуется использовать динамическое выделение памяти внутри некоторой функции, причём так, чтобы выделенная в ней память не потерялась, а объявленный на массив указатель нужным образом изменился. Для того чтобы сделать это, вы можете вы можете выбрать один из двух путей (выбрать один из двух стульев, так сказать):

1. передать в функцию указатель на имеющийся указатель на массив и работать внутри функции с двойным указателем

2. вернуть из функции указатель на выделенную область памяти и присвоить его текущему указателю на массив.

Выбирайте, и вперёд.

Обратите внимание, что программа должна работать до тех пор, пока пользователь не решит её закрыть. Каждое следующее действие, которое пользователь желает выполнить, должно выполняться над массивом с том состоянии, в котором он оказался после выполнения предыдущего действия.

Время на выполнение – 1 неделя

## Работа №3: Работа с циклическим списком

Необходимо написать программу, которая реализует хранение и простейший набор функций для обработки некоторой хранимой информации. Каждый хранимый элемент должен представлять собой структуру данных с необходимым набором полей, а все элементы должны представлять собой циклический список (односвязный или двусвязный – решайте сами). Программа должна иметь меню позволяющее выполнить следующие действия:

1. Добавить элемент

2. Удалить элемент

3. Вывести элементы

5. Сохранить данные в файл

6. Загрузить данные из файла

7. Выйти

При добавлении элемента необходимо спросить у пользователя значения всех необходимых полей структуры, заполнить ими структуру, и добавить её в список (куда хотите).

При удалении элемента необходимо спросить у пользователя по значению какого поля структуры необходимо выполнить удаление и, если таких элементов несколько – удалить их все. Номер варианта выбирается следующим образом: берёте свой номер по журналу, находите остаток от его деления на 5 и прибавляете 1.

Структуры данных для хранения:

Вариант 1: Фамилия студента, имя студента, коэффициент IQ

Вариант 2: Фамилия автора, название книги, год выпуска

Вариант 3: Название страны, площадь, население

Вариант 4: Марка автомобиля, модель, стоимость

Вариант 5: Название небесного тела, масса, средний радиус.

Время на выполнение – 2 недели

## Работа №4: Проверка возможности хода

Всем вам, конечно же, известна такая старая добрая головоломка как «пятнашки». Кому неизвестно – вот картинка, которая поможет понять о чём речь (все детали можно найти в Интернете, если они нужны):



Рабочее поле пятнашек представляет собой поле размером 4х4. В каждой клетке поля, кроме одной, находится какая-либо фишка. В процессе выполнения ходов игрок имеет право, не вынимая фишку из коробки, передвинуть её с текущего места на свободное.

В рамках работы необходимо написать программу, которая считает из одного файла одну комбинацию фишек на игровом поле, из другого файла – другую комбинацию фишек на игровом поле, после чего скажет (напишет на экране), можно ли за один ход попасть из комбинации из файла 1 в комбинацию из файла 2. Имена файлов с комбинациями необходим прочитать из командной строки. Номера фишек можно задать просто цифрами. Как задать пустую фишку – решайте сами (в примере она задана как 00, но делать так вовсе не обязательно).

Если аргументов в строке хватать не будет – необходимо вывести пользователю сообщение с детальным описанием того, как необходимо запускать программу и какие параметры ей требуются.

Пример 1:

file1.txt:

01 02 03 04

05 06 07 08

09 10 11 12

13 14 15 00

File2.txt:

01 02 03 04

05 06 07 08

09 10 11 00

13 14 15 12

Ответ – ход возможен.

Пример 2:

file1.txt:

01 02 03 04

05 06 07 08

09 10 11 12

13 14 15 00

File2.txt:

01 02 03 04

05 06 07 00

09 10 11 08

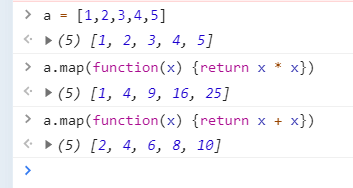
13 14 15 12

Ответ – ход невозможен.

Время на выполнение – 1 неделя

## Работа №5: Отображение

В большинстве современных языков программирования есть очень удобная функция map, с помощью которой можно выполнить отображение некоторого массива в другой массив по некоторому алгоритму. Алгоритм при этом задаётся функцией, которая и выполняет необходимое преобразование для каждого элемента массива. Вот пример её работы на языке JavaScript (я подозреваю что вы его не знаете, но код очень сложный, так что придётся вам его как-то понять):

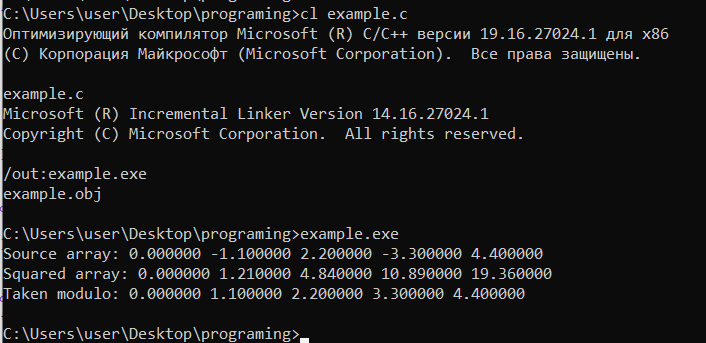


Комментарий к коду: есть исходный массив с именем а и значениями [1, 2, 3, 4, 5]. Нам необходимо каждый его элемент возвести в квадрат и вывести результат на экран, а затем каждый элемент исходного нетронутого массива а удвоить и так же вывести. Для того чтобы выполнить задуманное – мы вызываем функцию map и передаём ей аргумент – нашу собственную функцию (в данном случае без имени), которая и делает необходимые преобразования с каждым элементом массива и формирует из результатов новый массив (вот она: function(x) {return x \* x}).

Напишите аналог функции map на языке С. Функция должна принимать одномерный массив какого-то типа (выберите его сами), число элементов этого массива и указатель на функцию, которую необходимо применить к каждому из элементов. Функция должна возвращать указатель на НОВЫЙ массив такого же типа, как и исходный, в котором будут элементы исходного массива, к которым применена функция, переданная в качестве аргумента.

Программа должна содержать объявление исходного массива, требуемую функцию, вывод исходного массива на экран, а также вывод на экран результатов работы функции map для каких-либо двух функций по вашему желанию (например, для функции возведения в квадрат и для функции нахождения модуля числа).

Вот пример результатов компиляции и запуска программы для предложенных функций:



Время на выполнение – 1 неделя

## Работа №6 (вариант 1): Архипелаг

Использовать при решении этой задачи рекурсию запрещено!

В файле с произвольным именем задана прямоугольная карта произвольного размера, состоящая из «клеток», и олицетворяющая собой кусок поверхности планеты. В каждой может находиться либо суша (в этом случае значение в клетке будет «1»), либо вода (значение в клетке ­­- «0»).

Программа должна найти все острова на карте и вывести их площади на экран в порядке уменьшения. Минимальный размер острова – 1 клетка. Для островов б*о*льшего размера две клетки суши объединяются только по вертикали или по горизонтали, по диагонали – нет.

Пример карты в файле:

0 1 1 0 1 0

0 1 1 0 0 0

0 0 1 1 0 1

0 1 0 0 0 0

1 1 0 0 1 1

Результат для данной карты:

6 3 2 1 1

Время на выполнение – 2 недели

## Работа №6 (вариант 2): Экваториальный остров

Использовать при решении этой задачи рекурсию запрещено!

Представьте, что вы оказались на острове квадратной формы.

Ландшафт этого острова можно описать с помощью целочисленной матрицы размером M х N, каждый элемент которой задаёт высоту соответствующей области острова над уровнем моря (значения должны располагаться во внешнем файле, размер острова можно задать в программе).

Например, остров размером 3x3 может быть вот таким:

4 5 4

3 1 5

5 4 1

В сезон дождей остров полностью заливает водой, и в низинах скапливается вода. Низина - это такая область острова, клетки которой граничат с клетками, б*о*льшими по высоте (граничить не обязательно явно, вокруг низины могут быть и другие клетки низины, которые граничат с клетками, б*о*льшими по высоте). При этом диагональные соседи не учитываются, а уровень моря принимается за 0. В рамках работы необходимо для острова произвольного размера (чтобы вы не пугались – пусть он будет не больше чем 5х5 клеток) найти объём воды, который скапливается в его низинах

В приведённом выше примере на острове есть только одна низина - это клетка со значением 1 в середине острова (она граничит с клетками высотой 3, 5, 5 и 4).

Таким образом, после дождя высота клеток острова изменится и станет следующей:

4 5 4

3 3 5

5 4 1

Мы видим, что в данном примере высота низины изменилась с 1 до 3, после чего вода начала переливаться на соседние клетки, а затем — в море (всё что за пределами массива мы считаем морем куда всё сливается). Таким образом, для приведённого примера, общий объём воды, скопившейся на острове после сезона дождей - 2 кубические клетки.

Время на выполнение – 2 недели

## Работа №7: Команда MS-DOS

Написать свой аналог одной любой из команд MS-DOS из следующего списка:

- copy

- dir

- more

- move

Обратите внимание что все эти команды имеют те или иные ключи, каким-либо образом настраивающие их работу. Выберите из списка ключей команды любой 1 и реализуйте его поддержку. Реализовывать поддержку всех возможных ключей в рамках работы не требуется.

Время на выполнение – 2 недели