

1. What is the least number of operations necessary to sort an array of n arbitrary objects ?

В случае произвольных данных большинство самых быстрых алгоритмов требует около $O(n \log(n))$ операций

2. What is the most efficient data structure to support appending to the end of the set, removing last element from the set, as well as accessing or updating ith value ? Provide explanation and complexities.

Наиболее эффективная структура для хранения набора данных называется список. В данной структуре элементы хранят ссылки на другие элементы вместе с данными, а так же ссылка на первый и последний элемент списка. В зависимости от функциональных требований к структуре он может односвязным (хранится информация только о следующем или предыдущем элементе) либо полносвязным (хранится информация о следующем и предыдущем элементе). В данном случае в задаче требуется иметь возможность вставки, удаления или изменения произвольного элемента. В случае с односвязным списком, в зависимости от того данные о каком элементе хранятся будет высокая эффективность либо вставки, либо удаления, такие структуры так же называют Стек и Очередь. Но стек и очередь не эффективны на операциях вставки внутрь или и не могут быть одновременно эффективными при операциях и удаления и вставки. Причиной тому недостаток информации о предыдущем или о следующем элементе. Поэтому в данной задаче наиболее эффективной структурой будет полносвязный список (двунаправленный связный список), в такой структуре удаление, вставка, изменение одинаково эффективны.

3. What is the most efficient data structure to support inserting into a set as well as selection and deletion of a value from random position ? Provide explanation and complexities.

Наиболее эффективной структурой в данном случае является связанный список, т. к. операции поиска элементов по индексу, вставка и удаление в данной структуре наиболее эффективны.

Доступ	Поиск	Вставка	Удаление
$O(N)$	$O(N)$	$O(1)$	$O(1)$

4. What is virtual memory ? What is it used for ? Why it is necessary ?

Виртуальная память это метод управления памятью, при котором для освобождения быстрой и ограниченной оперативной памяти используется перенос не требующихся в данный момент данных на медленную память, например жесткий диск. Данный метод используется в большинстве операционных систем для решения проблемы распределения оперативной памяти, например многие приложения, т.к. это позволяет предоставить приложению больше памяти чем доступно оперативной памяти. Это позволяет избежать сбоев в работе приложений из за недостатка оперативной памяти путем увеличения нагрузки на медленную память.

5. Write an efficient program which given two sorted arrays $A[0..n]$ and $B[0..m]$ finds all values that are present in both array

<https://github.com/vpuhoff/Test-Data-Science/blob/master/5.ipynb>

6. Describe an algorithm to find a vertex with a highest degree in an undirected graph. Describe the complexity. You do not have to write a program or a full algorithm. The description is sufficient.

Степень вершины - это инцидентность вершины и ребра, так как граф неориентированный, то задача облегчается. Т.к. в условиях задачи ничего не сказано о входных данных приму, что изначально подается на вход матрица смежности:

СМЕЖНОСТИ:

ица	смежности	имеет			
	v_1	v_2	v_3	v_4	v_5
v_1	0	1	0	1	0
v_2	1	0	1	1	0
v_3	0	1	0	1	0
v_4	1	1	1	0	1
v_5	0	0	0	1	1

$A =$



И по этой матрице мы можем сразу легко дать ответ. Для каждой строки в матрице подсчитываем сумму элементов, строка с наибольшей суммой и будет представлять собой описание вершины с наибольшей степенью, например в матрице выше вершина V_4 будет иметь наибольшую степень.

7. Describe what is Object Oriented Programming ? What are the key concepts and characteristics ? Where is it used ? Write a small program demonstrating this idea.

Объектно-ориентированное программирование (ООП) — методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Основные принципы:

- абстрагирование (выделения важного)
- инкапсуляция (возможность описать команду «что делать», без одновременного уточнения как именно делать)
- наследование (организации родственных понятий: на каждом иерархическом шаге учитывать только изменения, не дублируя все остальное)
- полиморфизм (способность объекта использовать методы производного класса, который не существует на момент создания базового)

ООП реализовано в большинстве языков программирования и позволяет более точно описывать в рамках предметной области.

[Исходник](#)

8. Write a program that asks user's name, records it in memory, prints the number of times it saw the name since it was last started and goes back to asking user's name.

Исходный код:

```
static Dictionary<string, int> Memory = new Dictionary<string, int>();
static void Main(string[] args)
{
    do
    {
        Console.WriteLine("Введите имя:");
        string name = Console.ReadLine();
        if (!Memory.ContainsKey(name))
        {
            Memory.Add(name, 1);
            Console.WriteLine("Я впервые встретил данное имя.");
        }
        else
        {
            Memory[name]++;
            Console.WriteLine("Я встречал данное имя {0} раз.", Memory[name]);
        }
    } while (true);
}
```

9. Describe what is a 'global variable' in computer science domain ? What is it used for ? Write a small program demonstrating this idea.

В общем случае в программировании глобальной переменной называют переменную, областью видимости которой является вся программа. Пример:

```
static int GlobalVar = 1;
static void Main(string[] args)
{
    Func1();
    Func2();
    Console.ReadLine();
}
static void Func1()
{
    GlobalVar++;
    Console.WriteLine(GlobalVar);
}
static void Func2()
{
    GlobalVar = GlobalVar * 2;
    Console.WriteLine(GlobalVar);
}
```

10. What is the least number of operations necessary to sort an array of n integers from $0..m$?

$O(n)$ – (сравнение следующего с предыдущим, для случая когда исходный массив уже отсортирован)

$O(n \cdot \log(n))$ – для большинства алгоритмов наилучший результат

11. What is the most efficient data structure to support adding to a set as well as finding smallest value in the set and deleting a value added during i th iteration ? Provide explanation and complexities.

Наиболее эффективная структура для данной задачи будет Словарь (Dictionary), где ключем будет являться номер итерации. В таком случае поиск наименьшего элемента затребует $O(n)$ операций, вставка и удаление достаточно эффективны.

12. What is the most efficient data structure to support inserting in the beginning or the end of the set, removing element added during i th iteration and printing all elements in the right order in $O(n)$. Provide explanation and complexities.

Не совсем понятно что имеется в виду под «right order», т. к. не сказано о сортировке видимо имеется в виду, в каком порядке поступили данные в структуру, в таком же должны быть выведены. В таком случае можно воспользоваться немного измененной структурой связанного списка, добавив к элементу номер итерации в которой элемент был добавлен, в таком случае операция вставки в начало списка или в конец будут иметь эффективность $O(1)$, удаление

элемента, добавленного в I итерации будет произведено за $O(n)$, а вывод в правильном порядке будет осуществляться за $O(n)$. Добавление номера итерации необходимо, т. к. после удаления какого либо элемента внутри списка, если нет данной информации то определить точно в какой итерации был добавлен тот или иной элемент будет невозможно.

13. Explain what is 'segmentation fault' and when it occurs? Develop a program that demonstrates this idea.

Данный вид ошибки возникает при попытке обращения к недоступным для записи участкам памяти, либо при попытке изменения памяти запрещенным способом. Простейший способ ее вызвать:

```
void Main()
{
    Main();
}
```

Рекурсивный вызов функции приведет к переполнению стека и к данной ошибке.

14. Write a program that given an array $A[1...n]$ and a value S finds $0 < i < j < n$ such that $A[i] + A[j] = S$

Данную задачу можно решить путем перебора, либо путем представления задачи в форме СЛАУ и решения любым из методов, в данном случае я использовал графический метод решения.

[Исходный код](#)

15. Describe an algorithm that given a matrix described below determines whether matrix contains a value k . The input matrix $A[1...n, 1...n]$ has all rows and columns arranged in a non-descending order $A[i, j] < A[i, j+1], A[j, i] < A[j+1, i]$ for all $1 < i < n$ and $1 < j < n$.

Допустим массив заполняется по формуле $f(i, j)$, где $f(i, j) < f(i, j+1), f(j, i) < f(j+1, i)$ для всех $1 < i < n$ и $1 < j < n$.

В таком случае можно определить содержит ли массив число k выполнив сравнение:

Если $k \leq f(n, n)$, то массив может содержать данное число k , если $k > f(n, n)$ массив не может содержать числа k .

Однозначно утверждать наличие или отсутствие числа k для произвольного f невозможно, т. к. существуют f для которых k может не существовать. Например $f(i, j) = i^{10} + j^{10}$

Для построения алгоритма однозначно утверждающего наличие или отсутствие числа k нужно решить уравнение $f(i, j) = k$, если в результате i и j будут удовлетворять условиям задачи, то k существует.

16. Describe what is an 'instruction' in computer science domain ? How many instructions can be executed at a time ? Given a computer with one CPU (and one core), describe how modern systems achieve multitasking i.e. ability to execute multiple tasks at the same time.

Инструкция это минимальный фрагмент кода программы, исполняемый процессором. Количество одновременно выполняющихся инструкций можно определить по разному. Количество инструкций именно исполняющихся в текущий момент определяется числом ядер. Но количество инструкций находящихся в обработке процессором может превышать число ядер, т. к. современные процессоры используют конвейеры для одновременной обработки нескольких инструкций в рамках одного ядра. Инструкции в современных процессорах разделяются на микрооперации, которые выполняются независимо друг от друга. Для достижения возможности исполнения нескольких задач одновременно используются прерывание (аппаратное, либо программное), при котором происходит переключения контекста выполняемой задачи, то есть сохраняется текущее состояние вычисления, выгружается, загружается состояние из контекста другой задачи, производится исполнение и снова переключается на выполнение следующей задачи.

17. Write a program without using a built in square-root operator to compute square-root of a number that is known to have an integer square root.

У квадрата со стороной A площадь равна $S = A * A$. Из этого следует, что если нужно вычислить корень из S – нужно построить такой квадрат, что его площадь будет равна S , а длина стороны квадрата как раз и будет искомым корнем ($a = \sqrt{S}$).

```
static double Sqrt(double x, double eps)
{
    double height = 1;
    double width = x;
    do
    {
        height = (height + width) / 2;
        width = x / height;
    }
```

```
    } while (Math.Abs(width-height)>eps );  
    return height;  
}
```

Исходный код

18. Describe what is a garbage collection and how does it work ? What are its upsides and downsides ?

Реализация сборщика может сильно отличаться для каждого языка программирования или среды исполнения. Если обобщить сборщик мусора отслеживает объекты в памяти, на которые нет ссылок в программе, то есть те объекты к которым доступ осуществиться уже не может. Схему работы можно представить например так - при создании ссылки на объект в нем увеличивается значение счетчика ссылок, при удалении ссылки значение уменьшается, в таком случае можно удалить те объекты на которых значение счетчика равно 0.

Плюсы:

- Упрощение программирования
- Меньше утечек памяти
- Исключение появления ссылок на преждевременно удаленный объект
- Возможна выше скорость доступа к памяти (в перемещающем сборщике), за счет большей вероятности попадания данных в кэш процессора.
- Возможность использования финализаторов (возможность указать, что делать при удалении объекта)

Минусы:

- Меньше эффективность расходования ресурсов
- Сборщик мусора не панацея и в высоконагруженном ПО требуется внимательно следить за жизненным циклом объектов, иначе при неправильной реализации утечки памяти так же возможны
- В зависимости от реализации сборщика может быть медленнее выделение памяти или же освобождение
- Необходимость в среде исполнения, поддерживающей сборщик
- Поддержка со стороны языка программирования
- Возможные замедления в работе программы (некоторые операции в сборщике могут блокировать поток программы)

19. Provide pseudo-code or a program using programming language for any of the following sorting algorithms: merge sort, quick sort, heap sort. Explain best, average and worst case time complexity.

Сортировка слиянием (Merge sort)

Лучший	Средний	Худший	Вспомогательная сложность*
$O(n \log(n))$	$O(n \log(n))$	$O(n \log(n))$	$O(n)$

Метод оказывается крайне эффективным, однако требуется выделить дополнительную память.

Быстрая сортировка (Quicksort)

Лучший	Средний	Худший	Вспомогательная сложность*
$O(n \log(n))$	$O(n \log(n))$	$O(n^2)$	$O(n)$

Метод показывает высокую эффективность, при этом он не требует расходов на дополнительную память. Однако, в худшем случае алгоритм оказывается крайне медленным ($O(n^2)$).

Сортировка кучи(Heapsort)

Гарантировано $O(n \log n)$, при этом не требует дополнительной памяти.

1. Наилучшим из данных алгоритмов считаю **Heapsort**, т. к. он не требует дополнительной памяти и показывает хорошую производительность.
2. Средним считаю **Merge**, т. к. в контексте вопроса имеет значение производительность и требование к памяти можно не учитывать.
3. Худшим считаю **Quicksort**, т. к. сложность в худшем случае может достигать $O(n^2)$, что значительно хуже по сравнению с другими алгоритмами.

Исходный код (Heap sort)

20. What is the most efficient data structure to support all following operations: inserting into a set, finding an element by value, deleting an element by value. Provide explanation and complexities.

В данном случае наиболее эффективной структурой будет являться Хэш-таблица. Данная структура позволяет хранить пары ключ-значение и быстро выполнять перечисленные в задаче операции (выбор по значению, вставка и удаление элемента по значению).

21. What is the most efficient data structure to support all following operations: finding minimum value, finding and deleting an element by value. Provide explanation and complexities.

Структура данных наиболее эффективная в данном случае это двоичное дерево поиска, т. к. данные в нем хранятся в отсортированном виде и указанные в вопросе операции в данной структуре выполняются достаточно эффективно.

22. Write a program that given a positive integer N finds all such positive integers a and b that satisfy $a^2 + b^2 = N$

Из данного уравнения очевидно, что $a < \sqrt{N}$, т. к. иначе $a^2 > N$ и $b < \sqrt{N}$ по той же причине.

Для указанного уравнения так же можно выразить a и b, то есть получить равенство вида $a = \sqrt{N - b^2}$, то есть при известном значении b, значение a можно найти благодаря указанному равенству. Таким образом для поиска всех положительных целых чисел достаточно проверить целые числа от $0 < b < \sqrt{N}$ и проверить является ли a целым числом для данных значений числа b.

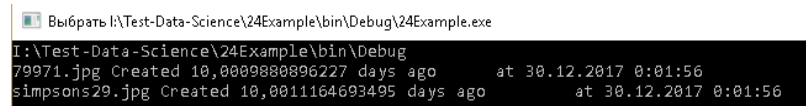
Исходный код

23. Write a program that determines if a given string is a palindrome. (A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward or forward)

Исходный код

24. Write a program using any technology that finds all files in a current directory that are older then 3 days.

Исходный код



```
Выбрать I:\Test-Data-Science\24Example\bin\Debug\24Example.exe
I:\Test-Data-Science\24Example\bin\Debug
79971.jpg Created 10,000,000,000,000 days ago at 30.12.2017 0:01:56
simpsons29.jpg Created 10,000,000,000,000 days ago at 30.12.2017 0:01:56
```

25. Given a program written using a language of your choice, describe how this program is executed ?

При программировании в среде .net компиляция и запуск приложений происходит следующим образом:

1. Код из любого .net языка преобразовывается в код, написанный на общем языке (Common intermediate language или CIL). Этот язык является языком низшего уровня, похожего по синтаксису на язык ассемблер.
2. После, этот код передаётся исполняющей среде (Common language runtime или CLR), которая извлекает код функций и методов из .net Framework
3. После этого конечный результат передаётся на исполнение

В случае с .net языком п.2-3 выполняются постоянно для запущенного приложения.