Минобрнауки России

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Факультет компьютерных и физико-математических наук

Кафедра прикладной математики и инфо­­­­­­­­­­рматики

Отчёт о прохождении

учебной практики

02.03.02 Фундаментальная информатика   
и информационные технологии

Выполнил: студент 1 курса   
`

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель: к. п. н., доцент Котельникова А. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Киров

2019 г.**Содержание**

Введение3

[Список задач 4](#_Список_задач)

[Описание решенных задач 6](#_Описание_решенных_задач)

[Список консультаций](#_Список_консультаций) 28

[Заключение 29](#_Заключение)

[Список литературы 30](#_Список_литературы)

[Приложения 31](bookmark://_Toc6456076#_Toc6456076)

Введение

Учебная практика проходила в ФГБОУ ВО «Вятский государственный университет», на кафедре прикладной математики и информатики факультета компьютерных и физико-математических наук с 04.02.2019 г. по 28.04.2019 г., количество недель: 12. Установочная конференция состоялась 04.02.2019 г.

Целью учебной практики является закрепление и углубление теоретической подготовки, приобретение практических навыков и компетенций в сфере профессиональной деятельности, в основном путём самостоятельного решения предусмотренных программой задач.

Форма проведения учебной практики: компьютерная (практикум по решению задач на ЭВМ).

В ходе прохождения практики необходимо реализовать компьютерные программы на языке программирования С++. Каждая задача сначала должна быть принята системой (acmp.ru), а потом нужно защитить предложенный алгоритм решения задачи и его программную реализацию в часы консультаций.

Для получения доступа к архиву задач была пройдена регистрация на сайте acmp.ru, регистрационное имя: Глызин Андрей Сергеевич.

Для реализации задач использовалась среда программирования Microsoft Visual C++.

# Список задач

Индивидуальный список задач из архива задач с сайта acmp.ru, полученный на установочной конференции:

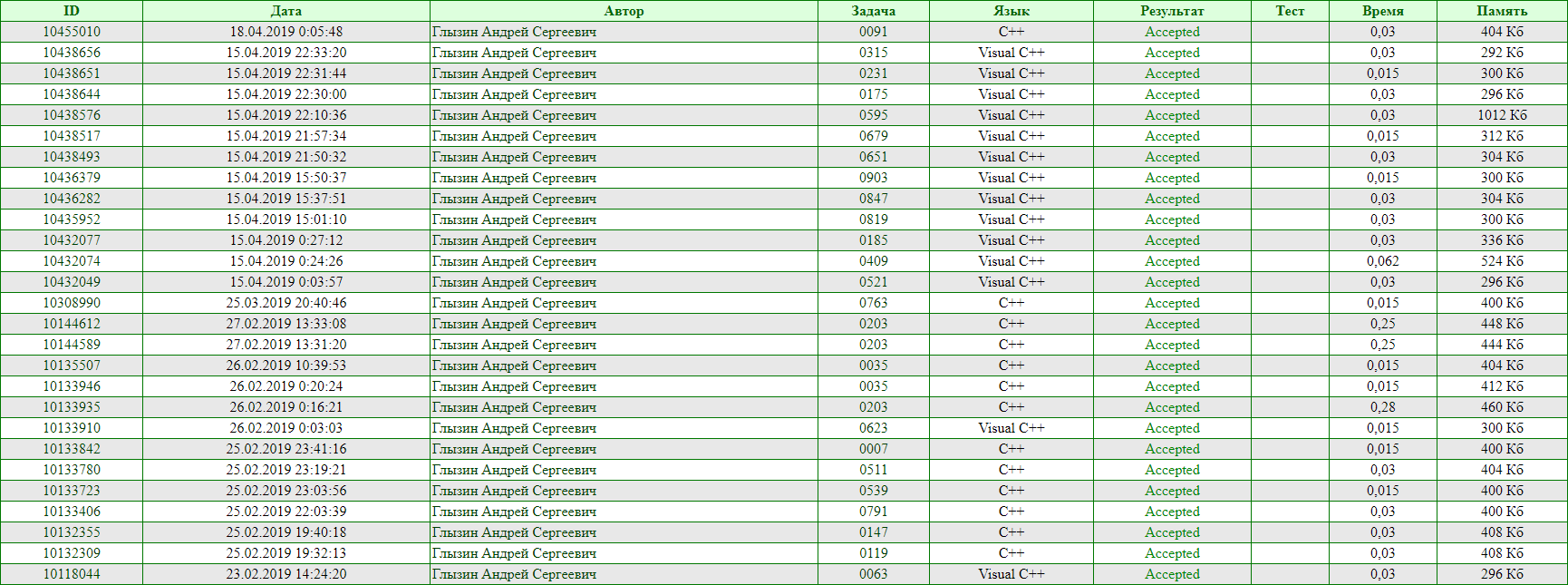
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № п/п | ID | Название | Тема | Разбор | Сложность | Решаемость |
| 1 | 0007 | Золото племени АББА | Длинная  арифметика |  | *30%* | 67% |
| 2 | 0035 | Конечные автоматы | Задачи для начинающих |  | 11% | 96% |
| 3 | 0063 | Загадка | Простая математика | Да | 18% | 94% |
| 4 | 0091 | Две последовательности | Сортировка и последовательности |  | 29% | 95% |
| 5 | 0119 | Сортировка времени | Сортировка и последовательности | Да | 23% | 92% |
| 6 | 0147 | Числа Фибоначчи | Целочисленная арифметика | Да | 16% | 94% |
| 7 | 0175 | Наручные часы | Математическое моделирование |  | 37% | 71% |
| 8 | 0203 | Сдвиг текста | Разбор строк |  | 31% | 62% |
| 9 | 0231 | Распаковка строки | Разбор строк |  | 25% | 82% |
| 10 | 0315 | Наименьшая система счисления | Задачи для начинающих |  | 26% | 75% |
| 11 | 0511 | Очередь | Задачи для начинающих |  | 20% | 63% |
| 12 | 0539 | Торт | Задачи для начинающих |  | 6% | 91% |
| 13 | 0595 | Слова | Разбор строк |  | 48% | 48% |
| 14 | 0623 | Снова Фибоначчи | Целочисленная арифметика |  | 23% | 76% |
| 15 | 0651 | Преобразование моноклеточных | Целочисленная арифметика |  | 29% | 78% |
| 16 | 0679 | Просто простые числа | Целочисленная арифметика |  | 40% | 93% |
| 17 | 0763 | Игра с ладьей | Задачи для начинающих |  | 19% | 76% |
| 18 | 0791 | Соседние клетки | Задачи для начинающих |  | 22% | 94% |
| 19 | 0819 | Прямоугольный параллелепипед | Задачи для начинающих |  | 10% | 91% |
| 20 | 0847 | Анаграмма-2 | Разбор строк |  | 28% | 86% |
| 21 | 0903 | Бисер | Задачи для начинающих |  | 2% | 97% |

# Описание решенных задач

В ходе прохождения практики мною были решены следующие задачи: 7, 35, 63, 91, 119, 147, 175, 203, 231, 315, 511, 539, 595, 623, 651, 679, 763, 791, 819, 847, 903, всего 21 задача. Каждая задача была реализована на языке программирования С++.

Детально будут описаны наиболее интересные с точки зрения решения и тестирования решения задачи 0847 и 0091.

На следующем скриншоте показаны принятые системой задачи.



1. ***0007 Золото племени АББА***

*(Время: 1 сек. Память: 16 Мб Сложность: 30%)*

Главный вождь племени Абба не умеет считать. В обмен на одну из его земель вождь другого племени предложил ему выбрать одну из трех куч с золотыми монетами. Но вождю племени Абба хочется получить наибольшее количество золотых монет. Помогите вождю сделать правильный выбор!

**Входные данные**

В первой строке входного файла INPUT.TXT записаны три натуральных числа через пробел. Каждое из чисел не превышает 10100. Числа записаны без ведущих нулей.

**Выходные данные**

В выходной файл OUTPUT.TXT нужно вывести одно целое число — максимальное количество монет, которые может взять вождь.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 5 7 3 | 7 |
| 987531 234 86364 | 987531 |
| 189285 283 4958439238923098349024 | 4958439238923098349024 |

***Описание решения:*** Для решения задачи необходимо сравнить три числа и вывести на экран большее из них. Используются три переменных типа string, их посимвольное сравнение происходит в отдельной функции.

Листинг программы приведен в Приложении 1.

1. ***0035 Конечные автоматы***

*(Время: 1 сек. Память: 16 Мб Сложность: 30%)*

Главный вождь племени Абба не умеет считать. В обмен на одну из его земель вождь другого племени предложил ему выбрать одну из трех куч с золотыми монетами. Но вождю племени Абба хочется получить наибольшее количество золотых монет. Помогите вождю сделать правильный выбор!

**Входные данные**

Первая строка входного файла INPUT.TXT содержит целое число k (1 ≤ k ≤ 10000) – количество конечных автоматов. Следующие k строк содержат по два целых числа ni (0 ≤ ni ≤ 1000) и mi (0 ≤ mi ≤ 26ni2) – число состояний и переходов i-го автомата

**Выходные данные**

Выходной файл OUTPUT.TXT должен состоять из k строк. На i-й строке выходного файла выведите одно число – нетривиальность i-го автомата.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 4 2 0 13 20 5 23 18 6 | 44344 48134 45699 49458 |
| 2 15 20 1000 26000 | 48767 1340237 |

***Описание решения:*** Для решения задачи необходимо вычислять значение по формуле исходя из исходных данных. Используется динамический массив переменных типа integer.

Листинг программы приведен в Приложении 2.

1. ***0063 Загадка***

*(Время: 1 сек. Память: 16 Мб Сложность: 18%)*

Петя и Катя – брат и сестра. Петя – студент, а Катя – школьница. Петя помогает Кате по математике. Он задумывает два натуральных числа X и Y (X,Y≤1000), а Катя должна их отгадать. Для этого Петя делает две подсказки. Он называет сумму этих чисел S и их произведение P. Помогите Кате отгадать задуманные Петей числа.

**Входные данные**

Входной файл INPUT.TXT содержит два натуральных числа S и P, разделенных пробелом.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите два числа Х и Y, загаданные Петей. Числа следует вывести в порядке неубывания своих значений, разделенные пробелом.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 4 4 | 2 2 |
| 987531 234 86364 | 987531 |

***Описание решения:*** Для решения задачи необходимо совершить перебор чисел, и при соблюдении всех условий вывести результат на экран. Используется структура, которая содержит две переменных типа integer, для удобства.

Листинг программы приведен в Приложении 3.

1. ***0091 Две последовательности***

*(Время: 1 сек. Память: 16 Мб Сложность: 29%)*

Определим последовательности an и bn следующим образом: a1 = 2, a2 = 3, a3 = 4, a4 = 7, a5 = 13, an = bn−1 + bn−3, n > 5, bn — последовательность чисел, не входящих в an, записанных в возрастающем порядке.

Таким образом, последовательность an будет выглядеть следующим образом: 2, 3, 4, 7, 13, 15,..., а последовательность bn – 1, 5, 6, 8, 9, 10,....

Ваша задача состоит в том, чтобы найти an и bn.

**Входные данные**

Входной файл содержит целое число n (1 ≤ n ≤ 10000).

**Выходные данные**

В первой строке выходного файла выведите an, во второй – bn.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 4 | 7  8 |
| 10 | 25 16 |
| 6578 | 19731 9868 |

***Описание решения:*** Для решения задачи необходимо продолжить последовательности и найти n-х членов этих последовательностей. Для этого используется вектор bool, который определяет принадлежность переменной к последовательности a. Так же в ходе выполнения программы члены этах последовательностей не записываются в массив, и храниться всего лишь 4 значения b(i) b(i-1) b(i-2) b(i-3).

Создаётся вектор логических переменных, количество которых равно 3\*n/2+100. Это соотношение находиться очень просто, из данных условия. Если мы поделим 19731 на 6578 (элемент последовательности a делённый и свой номер), то получим число очень близкое к двум - 2.99954. Тоже самое можно проделать с n-ым числом последовательности b - 9868/6578 = 1.50015. Можно исходя из этого получить примерное значение n-го элемента последовательности b по формуле 3\*n/2+100, а 100 прибавляется чтобы задать начальные значения элементов вектора. Далее задаются изначальные значения “true”, для элементов с индексами элементов последовательности а. Так же создаются три переменных со значениями последних элементов последовательности b. В последующий цикл мы можем не попасть из-за его условия. Он предназначен для нахождения элементов, индексы которых меньше 7. В этом случае выводим на экран значения из массивов a и b, которые описанных в конце кода. В ином случае мы попадаем в цикл, в ходе которого будет определяться значение a[i] и b[i].

Пример. При вводе 4 из первого теста acmp.ru в цикл мы не попадём, а значения попадут на экран из массива в конце кода.

При вводе 10 мы попадаем в цикл сразу определяем a[7]=18, b[7]=11, следующее выполнение цикла - a[8]=20, b[8]=12. При i = 9 так же находиться значение a[9]=22, но а значение b[9]=13 нам не подходит, поскольку оно уже есть в последовательности a, поэтому прибавляем единицу и получаем 14. Та же самая ситуация происходит в следующем выполнении цикла и мы получаем значения 25 и 16. Они и будут ответом для числа 10.

Листинг программы приведен в Приложении 4.

1. ***0119 Сортировка времени***

*(Время: 1 сек. Память: 16 Мб Сложность: 23%)*

Требуется выполнить сортировку временных моментов, заданных в часах, минутах и секундах.

**Входные данные**

Во входном файле INPUT.TXT в первой строке записано число N (1 ≤ N ≤ 100), а в последующих N строках N моментов времени. Каждый момент времени задается 3 целыми числами - часы (от 0 до 23), минуты (от 0 до 59) и секунды (от 0 до 59).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите моменты времени, упорядоченные в порядке неубывания без ведущих нулей.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 4 10 20 30 7 30 00 23 59 59 13 30 30 | 7 30 0 10 20 30 13 30 30 23 59 59 |

***Описание решения:*** Для решения задачи нужно последовательно записывать в массив определённые моменты. После этого нужно перевести время в секунды и выполнить сортировку. Используется структура из трех переменных типа integer - часы, минуты, секунды.

Листинг программы приведен в Приложении 5.

1. ***0147 Числа Фибоначчи***

*(Время: 1 сек. Память: 16 Мб Сложность: 16%)*

Последовательностью Фибоначчи называется последовательность чисел a0, a1, ..., an, ..., где  
a0 = 0, a1 = 1, ak = ak-1 + ak-2 (k > 1).

Требуется найти N-е число Фибоначчи.

**Входные данные**

Во входном файле INPUT.TXT записано целое число N (0 ≤ N ≤ 30).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите N-е число Фибоначчи.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 7 | 13 |

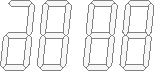
***Описание решения:*** Для решения задачи необходимо найти n-e число Фибоначчи по формуле, указанной в описании. Используется рекурсивная функция.

Листинг программы приведен в Приложении 6.

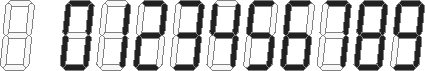
1. ***0175 Наручные часы***

*(Время: 1 сек. Память: 16 Мб Сложность: 37%)*

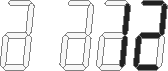
Вы приобрели новые электронные наручные часы с жидкокристаллическим дисплеем. Дисплей отображает часы и минуты с помощью четырех элементов, каждый из которых отображает одну цифру.



Три из них состоят из семи полосок, каждая из которых может быть либо белой (неотличимой от фона), либо черной. Вид такого элемента и отображаемые им цифры показаны на рисунке:



Четвертый элемент предназначен для отображения старшей цифры часа. Если она равна нулю, то элемент полностью неактивен (все полоски белые), иначе показывается соответствующая цифра. Вот как выглядит этот элемент с цифрами:



Вам хочется проверить: все ли в порядке с новым приобретением, а именно, нет ли таких полосок в каком-либо из элементов, которые либо всегда белые, либо всегда черные. Вы хотите начать проверку в некоторое начальное время. Требуется определить, сколько Вам потребуется минут для убеждения в исправности часов.

**Входные данные**

В первой строке входного файла INPUT.TXT находится время начала проверки в формате HH:MM. Часы и минуты записаны с лидирующими нулями, если таковые имеются. (00 ≤ HH ≤ 23, 00 ≤ MM ≤ 59).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите минимальное число минут, необходимое для проверки Ваших часов, если она началась в заданное время.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 00:00 | 1200 |
| 02:39 | 1041 |

***Описание решения:*** Для решения задачи необходимо выявить закономерность исходя из полосок, которые загораются или гаснут в зависимости от цифры. Правая часть часов (минуты). Может полностью пройти проверку за один час. Следовательно решение упирается только в часы. В конце решения найденное время переводиться в минуты. Используется переменная string и две целочисленные переменные.

Листинг программы приведен в Приложении 7.

1. ***0203 Сдвиг текста***

*(Время: 0,5 сек. Память: 16 Мб Сложность: 31%)*

Мальчик Кирилл написал однажды на листе бумаги строчку, состоящую из больших и маленьких английских букв, а после этого ушел играть в футбол. Когда он вернулся, то обнаружил, что его друг Дима написал под его строкой еще одну строчку такой же длины. Дима утверждает, что свою строчку он получил циклическим сдвигом строки Кирилла направо на несколько шагов (циклический сдвиг строки abcde на 2 позиции направо даст строку deabc). Однако Дима известен тем, что может случайно ошибиться в большом количестве вычислений, поэтому Кирилл в растерянности - верить ли Диме? Помогите ему!

По данным строкам выведите минимально возможный размер сдвига вправо или -1, если Дима ошибся.

**Входные данные**

Первые две строки входного файла INPUT.TXT содержат строки Кирилла и Димы соответственно. Строки состоят только из английских символов. Длины строк одинаковы, не превышают 10000 и не равны 0.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите единственное число - ответ на поставленную задачу.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| abcde deabc | 2 |

***Описание решения:*** Для решения задачи необходимо выполнить посимвольное сравнение со сдвигом. Используется класс <string> и циклы for, while.

Листинг программы приведен в Приложении 8.

1. ***0231 Распаковка строки***

*(Время: 1 сек. Память: 16 Мб Сложность: 25%)*

Будем рассматривать только строчки, состоящие из заглавных английских букв. Например, рассмотрим строку AAAABCCCCCDDDD. Длина этой строки равна 14. Поскольку строка состоит только из английских букв, повторяющиеся символы могут быть удалены и заменены числами, определяющими количество повторений. Таким образом, данная строка может быть представлена как 4AB5C4D. Длина такой строки 7. Описанный метод мы назовем упаковкой строки.

Напишите программу, которая берет упакованную строчку и восстанавливает по ней исходную строку.

**Входные данные**

Входной файл INPUT.TXT содержит одну упакованную строку. В строке могут встречаться только конструкции вида nA, где n — количество повторений символа (целое число от 2 до 99), а A — заглавная английская буква, либо конструкции вида A, то есть символ без числа, определяющего количество повторений. Строка содержит от 1 до 80 символов.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите восстановленную строку. При этом строка должна быть разбита на строчки длиной ровно по 40 символов (за исключением последней, которая может содержать меньше 40 символов).

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 3A4B7D | AAABBBBDDDDDDD |
| 22D7AC18FGD | DDDDDDDDDDDDDDDDDDDDDDAAAAAAACFFFFFFFFFF FFFFFFFFGD |
| 95AB | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAB |
| 40AB39A | AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA BAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA |

***Описание решения:*** Для решения задачи необходимо выводить на экран определённое количество символов исходя из цифры перед этими символами, так же написана отдельная функция для переноса строки на 40ом символе. Используется класс <string> и функции для проверки символов на их принадлежность к буквам алфавита и цифрам.

Листинг программы приведен в Приложении 9.

1. ***0315 Наименьшая система счисления***

*(Время: 1 сек. Память: 16 Мб Сложность: 26%)*

Известно, что основанием позиционной системы счисления называют количество различных символов, используемых для записи чисел в данной системе счисления. Также известно, что любое число x в b-ичной системе счисления имеет вид x=a0∙b0+a1∙b1+…+an∙bn, где b ≥ 2 и 0 ≤ ai < b.

Для записи чисел в b-ичной системе счисления, где b ≤ 36, могут быть использованы первые b символов из следующего списка 0,1,…, 9, A, B, …, Z. Например, для записи чисел в троичной системы используются символы 0, 1, 2, а в двенадцатеричной - 0,1,…, 9, A, B.

Требуется написать программу, которая по входной строке S определит, является ли данная строка записью числа в системе счисления, с основанием не большим 36, и, если является, определит минимальное основание этой системы счисления.

**Входные данные**

Входной файл INPUT.TXT содержит в единственной строке входную непустую строку. Длина строки не превышает 255. Все символы строки имеют коды от 32 до 127.

**Выходные данные**

Выходной файл OUTPUT.TXT должен содержать одно число. Если строка является записью числа в некоторой системе счисления, то нужно вывести минимальное основание такой системы счисления. Иначе вывести -1.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 123 | 4 |
| ABCDEF | 16 |
| AD%AF | -1 |
| 03025 | 6 |
| abc | -1 |

***Описание решения:*** Для решения задачи необходимо найти максильный номер символа, который попадает под ограничения 0,1,…, 9, A, B, …, Z. Используется строковый тип данных.

Листинг программы приведен в Приложении 10.

1. ***0511 Очередь***

*(Время: 1 сек. Память: 16 Мб Сложность: 20%)*

Студент Василий живет в общежитии. Отделение банка, в котором он производит оплату за проживание, имеет всего две кассы, поэтому почти всегда длинная очередь к ним. Первая касса открывается в 8.00, вторая – в 8.05. Последний клиент будет принят в 20.00. Очередь единая, и очередной клиент обслуживается, как только освобождается одна из касс. На обслуживание одного клиента уходит ровно 10 минут. Василий приходит ровно в 8.00 и видит, сколько человек стоит перед ним. Требуется определить, сколько времени ему придется простоять в очереди, и вообще обслужат ли его сегодня.

**Входные данные**

Входной файл INPUT.TXT содержит единственное натурально число K – номер Василия в очереди (K < 250).

**Выходные данные**

В выходной файл OUTPUT.TXT выводится строка «NO», если Василий сегодня заплатить уже не успеет, и время его ожидания (в формате «X Y», где X – количество целых часов, которые простоит в очереди Василий, и Y – количество минут), если все же успеет заплатить.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 1 | 0 0 |
| 20 | 1 35 |
| 235 | NO |

***Описание решения:*** Для решения задачи необходимо проверить номер Василия в очереди. Есть четыре варианта: Он идёт сразу (его номер <=2), Попадает в первую кассу (остаток от деления его номера на 2 равен единице), Попадает во вторую кассу (в ином случае) и не успевает вообще, если его номер больше 145. Остаётся лишь учесть разницу в 5 минут во втором и третьем случае, когда до него удачно доходит очередь. Используется целочисленный тип данных.

Листинг программы приведен в Приложении 11.

1. ***0539 Торт***

*(Время: 1 сек. Память: 16 Мб Сложность: 6%)*

На свой день рождения Петя купил красивый и вкусный торт, который имел идеально круглую форму. Петя не знал, сколько гостей придет на его день рождения, поэтому вынужден был разработать алгоритм, согласно которому он сможет быстро разрезать торт на N равных частей. Следует учесть, что разрезы торта можно производить как по радиусу, так и по диаметру.

Помогите Пете решить эту задачу, определив наименьшее число разрезов торта по заданному числу гостей.

**Входные данные**

Входной файл INPUT.TXT содержит натуральное число N – число гостей, включая самого виновника торжества (N ≤ 1000).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите минимально возможное число разрезов торта.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 2 | 1 |
| 3 | 3 |

***Описание решения:*** Для решения задачи необходимо определить чётность количества гостей. Используется целочисленная переменная и оператор получения остатка от деления.

Листинг программы приведен в Приложении 12.

1. ***0595 Слова***

*(Время: 0,5 сек. Память: 16 Мб Сложность: 48%)*

Для шифрования слов с ними можно производить множество различных операций. Например, интересна такая операция: первые несколько букв заданного слова приписываются к его концу в обратном порядке, после чего удаляются из начала слова. При этом слово a1a2 … akak+1 … an переходит в слово ak+1 … anakak-1 … a1 (число k выбирается в диапазоне от 0 до n).

Для двух заданных слов требуется определить, можно ли применением описанной операции преобразовать первое слово во второе.

**Входные данные**

Входной файл INPUT.TXT состоит из двух строк - на первой из них записано исходное слово, а во второй - предполагаемый результат. Слова состоят из строчных и заглавных английских букв. Длины строк не превышают 50000 символов.

**Выходные данные**

В первой строке выходного файла OUTPUT.TXT выведите "Yes", если преобразование возможно, и "No", если нет. В случае положительного ответа во второй строке выведите k - длину перемещаемой части исходного слова k (из всех таких k выберите минимальный).

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| wpwdwpw wdwpwpw | Yes 2 |
| dWOddWd dOdWdWd | No |

***Описание решения:*** Для решения задачи нужно найти кусочек начала строки совпадающий с концом другой строки. Исходя из условия нам нужен больший кусочек из возможных. Используется строковый тип данных и вектор числовых переменных.

Листинг программы приведен в Приложении 13.

1. ***0623 Снова Фибоначчи***

*(Время: 2 сек. Память: 16 Мб Сложность: 23%)*

Вам наверняка знакомы числа Фибоначчи: 1, 1, 2, 3, 5, 8, 13, 21... Они определяются рекуррентным соотношением: Fn = Fn-1 + Fn-2, F0 = F1 = 1.

Требуется найти последнюю цифру n-го числа Фибоначчи.

**Входные данные**

Во входном файле INPUT.TXT содержится одно целое число n (0 ≤ n ≤ 108).

**Выходные данные**

В выходной файл OUTPUT.TXT необходимо вывести одно число - последнюю цифру числа Fn.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 1 | 1 |
| 5 | 8 |

***Описание решения:*** Для решения задачи достаточно воспользоваться свойством этих чисел. Их последние цифры образуют переодическую последовательность с периодом 60. Используются переменные типа long long и оператор взятия остатка от деления.

Листинг программы приведен в Приложении 14.

1. ***0651 Преобразование моноклеточных***

*(Время: 1 сек. Память: 16 Мб Сложность: 29%)*

Как великолепна страна Байтландия! В ней есть цветущие леса, прозрачные реки, кисельные берега… Но речь пойдет не о них. Уже много лет в Байтландии функционирует НИИ “Цитологии и генетики”. В нем выводятся новые формы жизни. Недавно ученым этого НИИ удалось разработать принципиально новый вид организмов. Особенностью этих организмов является то, что они состоят из большого количества однотипных клеток, то есть являются моноклеточными.

Правительство Байтландии заинтересовалось новой разработкой и сделало заказ на производство двух моноклеточных организмов, в каждом из которых должно быть по M клеток. За несколько дней до сдачи проекта было обнаружено, что в одном из организмов получается не M клеток, а N. На какой из стадий разработки была допущена ошибка неизвестно, но положение надо исправлять!

Сотрудниками НИИ было принято решение о преобразовании моноклеточного с N клетками в моноклеточное с M клетками. Для этого в экстренном режиме было разработано два типа вещества:

Вещество, которое делит клетки моноклеточного организма, т.е. каждая клетка делится на P частей. В результате количество клеток умножается на P, где P – простое число.

Вещество, объединяющее клетки. Клетки организма объединяются в группы по T штук, где Т также простое число. Далее каждая группа клеток объединяется в одну клетку. В результате общее количество клеток делится на T. При этом T выбирается таким, чтобы деление происходило без остатка.

Отметим, что натуральное число называется простым, если оно имеет только два натуральных делителя – это единица и само число.

Серьезным недостатком этих веществ является их высокая стоимость. В соответствии с этим требуется преобразовать моноклеточное с N клетками в моноклеточное с M клетками за минимальное количество операций. За одну операцию к моноклеточному можно применить одно вещество из двух заданных типов. Помогите НИИ “Цитологии и генетики” разрешить эту непростую задачу!

**Входные данные**

В первой строке входного файла INPUT.TXT заданы два натуральных числа N и M (1 ≤ N, M ≤ 109) разделенные одиночным пробелом.

**Выходные данные**

Единственная строка выходного файла OUTPUT.TXT должна содержать целое число – минимальное количество операций, необходимое для преобразования моноклеточного организма с N клетками в моноклеточный организм с M клетками.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 2 36 | 3 |
| 32768 3 | 16 |
| 1434 1434 | 0 |

***Описание решения:*** Для решения задачи необходимо найти сумму количества делителей от входных значений поделённых на их наибольший общий делитель. Используется целочисленный тип данных и две функции - Нахождение количества делителей и нахождение НОД.

Листинг программы приведен в Приложении 15.

1. ***0679 Просто простые числа***

*(Время: 1 сек. Память: 16 Мб Сложность: 40%)*

Дано натуральное число N. Представить его в виде суммы простых натуральных чисел так, чтобы произведение этих слагаемых было максимально.

**Входные данные**

В единственной строке входного файла INPUT.TXT записано одно натуральное число N (2 ≤ N ≤ 2 000 000 000).

**Выходные данные**

В единственную строку выходного файла OUTPUT.TXT нужно вывести простые числа по возрастанию с указанием их количества при разложении, т.е.: <число> <количество>.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 5 | 2 1 3 1 |
| 30 | 3 10 |

***Описание решения:*** Для решения задачи необходимо выразить число через минимальные, простые, натуральные числа - 2, 3. Используются две целочисленные переменные.

Листинг программы приведен в Приложении 16.

1. ***0763 Игра с ладьей***

*(Время: 1 сек. Память: 16 Мб Сложность: 19%)*

На бесконечной вправо и вверх шахматной доске находится ладья. Два игрока передвигают ее по очереди. За один ход разрешено сдвинуть ладью вниз или влево на произвольное (ненулевое) количество клеток так, чтобы ладья не покинула доску. Цель игры – переместить ладью в левый нижний угол, то есть клетку с координатами (1,1). Известно, что оба игрока придерживаются оптимальной стратегии. Игрок №1 ходит первым, при этом он обязан совершить хотя бы один ход. Если первый ход сделать нельзя, то определить победителя также невозможно. Требуется написать программу, которая найдет номер победившего игрока, либо определит, что этого сделать нельзя.

**Входные данные**

Входной файл INPUT.TXT содержит два натуральных числа, разделенных пробелами: X и Y – координаты ладьи перед первым ходом (X,Y ≤ 109).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите одно число – номер победившего игрока. Если победителя определить невозможно, то следует вывести 0.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 1 1 | 0 |
| 1 6 | 1 |

***Описание решения:*** Для решения задачи необходимо исходные координаты фигуры. Если она попадает на диагональ, то тогда выиграет второй игрок, т.к. первому необходимо сделать не нулевой ход, а второй всегда будет возвращать фигурку на диагональ. В остальных случаях выигрывает первый игрок. Используются две целочисленные переменные.

Листинг программы приведен в Приложении 17.

1. ***0791 Соседние клетки***

*(Время: 1 сек. Память: 16 Мб Сложность: 22%)*

Клетки шахматной доски пронумерованы числами от 1 до 64 по строкам слева направо и снизу вверх. Напишите программу, которая по заданному номеру клетки определяет номера всех клеток, имеющих с ней общую сторону.

**Входные данные**

Входной файл INPUT.TXT содержит одно целое число от 1 до 64.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите через пробел в порядке возрастания номера всех клеток, имеющих с заданной общую сторону.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 2 | 1 3 10 |
| 64 | 56 63 |

***Описание решения:*** Для решения задачи нужно продумать все возможные варианты. Используется целочисленная перменная и оператор switch.

Листинг программы приведен в Приложении 18.

1. ***0819 Прямоугольный параллелепипед***

*(Время: 1 сек. Память: 16 Мб Сложность: 10%)*

*Прямоугольный параллелепипед* - это объемная фигура, у которой шесть граней, и каждая из них является прямоугольником. Моделями прямоугольного параллелепипеда служат классная комната, кирпич, спичечная коробка. Длины трех ребер прямоугольного параллелепипеда, имеющих общий конец, называют его измерениями. На приведенном рисунке измерения представлены ребрами *AB*, *BC* и *BF* с общим концом в точке *B*, а значения измерений равны их длинам *a*, *b* и *c* соответственно.

По заданным измерениям прямоугольного параллелепипеда Вам необходимо определить площадь его поверхности и объем.

**Входные данные**

Единственная строка входного файла INPUT.TXT содержит разделенные пробелом три натуральных числа A, B и С – измерения параллелепипеда, каждое из измерений не превышает 106.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите через пробел два целых числа: площадь поверхности и объем заданного параллелепипеда.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 1 1 1 | 1 6 |
| 2 3 4 | 52 24 |

***Описание решения:*** Для решения задачи считать с клавиатуры значения сторон параллелепипеда и вычислить площадь поверхности и объём. Используется целочисленные переменные типа long long.

Листинг программы приведен в Приложении 19.

1. ***0847 Анаграмма-2***

*(Время: 1 сек. Память: 16 Мб Сложность: 28%)*

Катя и Таня играли в слова. Одна из девочек называла слово на английском языке, вторая должна найти анаграмму. Анаграмма – это слово, полученное из другого слова путем перестановки всех без исключения букв первого слова.

Написать программу, которая проверит, правильно ли девочки создают анаграммы.

**Входные данные**

Входной файл INPUT.TXT содержит два слова на английском языке в нижнем регистре, разделенные пробелом. Каждое слово содержит от 1 до 20 символов.

**Выходные данные**

В выходной файл OUTPUT.TXT выведите «YES», если анаграмма подобрана правильно, иначе выведите «NO».

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| post stop | YES |
| post pots | NO |
| aabb aabb | NO |

***Описание решения:*** Для решения задачи необходимо отсортировать слова по буква и проверить, все ли буквы в слове потерпели перестановку. Используются переменные типа string, char и bool. Сначала читаются два слова с клавиатуры в переменные строкового типа. Затем сравниваются количество символов в этих переменных, если их длинны совпадают, то затем каждое слово по очереди сортируется в алфавитном порядке. Если после сортировки значения переменных совпадают, то выводим “YES”, иначе выводим “NO”.

Разберём первый пример, предложенный на acmp.ru.  
Переменные будут равны “post” и “stop”. Количество их символов одинаково и равно четырём. После сортировки а алфавитном порядке мы получим набор символов “opst” в этих переменных, результат сравнения этих переменных будет истинным, соответственно на экран будет выведено сообщение “YES”

Листинг программы приведен в Приложении 20.

1. ***0903 Бисер***

*(Время: 1 сек. Память: 16 Мб Сложность: 2%)*

В шкатулке хранится разноцветный бисер (или бусины). Все бусины имеют одинаковую форму, размер и вес. Бусины могут быть одного из N различных цветов. В шкатулке много бусин каждого цвета.

Требуется определить минимальное число бусин, которые можно не глядя вытащить из шкатулки так, чтобы среди них гарантированно были две бусины одного цвета.

**Входные данные**

Входной файл INPUT.TXT содержит одно натуральное число N - количество цветов бусин (1 ≤ N ≤ 109).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите ответ на поставленную задачу.

Пример

|  |  |
| --- | --- |
| INPUT.TXT | OUTPUT.TXT |
| 3 | 4 |

***Описание решения:*** Для решения задачи необходимо прибавить к исходному числу единицу и мы получим ответ.  
Если на вход подаётся цифра два, то мы имеем два различных цвета. Допустим красный - К и синий - С.

Предположим, что нам нужно достать три бусины, чтобы получить гарантированно два разных цвета. Исходя из этого получаются варианты (перестановка не играет роли):

CCC

KKK

KCC

KKC  
В каждом из этих вариантов присутствует хотя бы две бусины одинакового цвета. Больше брать уже нет смысла, но взять меньшее количество можно попробовать:

CC

KK

KC  
Как видим, нашёлся вариант KC, где присутствие двух бусин одинакового цвета ложно.  
Используется целочисленная переменная.

Листинг программы приведен в Приложении 21.

# Список консультаций

За период прохождения практики были посещены следующие консультации.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Дата | Содержание | Сданные задачи |
|  | 27.02.2019 | Защита решений задач | 0007, 0035, 0063,  0119, 0147, 0203, 0511, 0539, 0623,  0791 |
|  | 18.04.2019 | Защита решений задач | 0903, 0847, 0819, 0763, 0679, 0651, 0595, 0315, 0231,  0175, 0091 |
|  | 27.04.2019 | Заключительная конференция |  |

# Заключение

Учебная практика способствовала закреплению и углублению теоретической подготовки, приобретению практических навыков и компетенций в сфере профессиональной деятельности путём самостоятельного решения задач.

В ходе практики была решена 21 задача, реализованных на языке программирования С++. Листинги всех составленных программ приведены в приложении.

Наиболее интересным оказалось получение нового опыта и знаний при решении задач.

Наиболее сложным оказались задачи тесно связанные с математикой.

# Список литературы

1. Ахо А., Хопкрофт Дж., Ульман Дж. Структуры данных и алгоритмы. – М.: Вильямс, 2010.
2. Вирт Н. Алгоритмы и структуры данных. М.: ДМК, 2009.
3. Гагарина, Л. Г. Алгоритмы и структуры данных [Электронный ресурс]: учебное пособие / Л. Г. Гагарина, В. Д. Колдаев. – М.: Финансы и статистика, 2009. – 304 с. (режим доступа: <http://biblioclub.ru/index.php?page=book&id=225965>).
4. Кнут Д. Искусство программирования. Том 1. Основные алгоритмы. – 3-е изд. – М.: Вильямс, 2011.
5. Кнут Д. Искусство программирования. Том 3. Сортировка и поиск. – 2-е изд. – М.: Вильямс, 2000.
6. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы. Построение и анализ. – М.: Вильямс, 2011.
7. Окулов С. М. Основы программирования [Текст] / С. М. Окулов. – 6-е изд., перераб. – М.: БИНОМ. Лаборатория Знаний, 2012. – 336 с.
8. Окулов С. М. Программирование в алгоритмах [Текст] / С. М. Окулов. – 3-е изд. – М.: БИНОМ. Лаборатория знаний, 2012. – 383 с.
9. Окулов С. М. Алгоритмы обработки строк [Электронный ресурс] / С. М. Окулов. – М.: БИНОМ. Лаборатория знаний, 2012. – 256 с. (режим доступа: <http://biblioclub.ru/index.php?page=book&id=222847>).
10. Окулов С. М., и др. Задачи по программированию [Текст] / С. М. Окулов, Т. В. Ашихмина, Н. А. Бушмелева, М. А. Корчемкин, Е. В. Разова, Р. В. Шарыгин; под ред. С. М. Окулова. – М.: БИНОМ. Лаборатория знаний, 2014, 823 с.
11. Окулов, С. М. Дискретная математика. Теория и практика решения задач по информатике [Текст] : учеб. пособие / С. М. Окулов. – М.: БИНОМ. Лаборатория знаний, 2013. – 422 с.

Приложения

**Приложение 1. Задача 0007. Золото племени АББА**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

string q(string s1, string s2)

{

**int** i = 0;

**if** (s1.length() > s2.length()) **return** s1;

**if** (s1.length() < s2.length()) **return** s2;

**while** (**true**)

{

**if** (s1[i] > s2[i])

{

**return** s1;

**return** 0;

}

**if** (s1[i] < s2[i])

{

**return** s2;

**return** 0;

}

i++;

}

**return** s1;

}

**int** main()

{

string s1, s2, s3;

cin >> s1 >> s2 >> s3;

cout << q(s1, q(s2, s3));

**return** 0;

}

**Приложение 2. Задача 0035. Конечные автоматы**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**int** k;

cin >> k;

**int**\* A = **new** **int**[k];

**for** (**int** i = 0; i < k; i++)

{

**int** n, m;

cin >> n >> m;

A[i] = 19 \* m + (n + 239)\*(n + 366) / 2;

}

**for** (**int** i = 0; i < k; i++)

{

cout << A[i]<<endl;

}

**return** 0;

}

**Приложение 3. Задача 0063. Загадка**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**struct** point {

**int** x;

**int** y;

};

point f(**int** s, **int** p)

{

point result;

**for** (**int** x = 0; x <= 1000; x++)

{

**for** (**int** y = 0; y <= 1000; y++)

{

**if** (x + y == s && x\*y == p)

{

s = x; p = y;

**break**;

}

}

}

**if** (s < p) {

result.x = s;

result.y = p;

}

**else**

{

result.x = p;

result.y = s;

}

**return** result;

}

**int** main()

{

**int** s, p;

cin >> s >> p;

cout << f(s, p).x << " " << f(s, p).y;

**return** 0;

}

**Приложение 4. Задача 0091. Две последовательности**

#include <iostream>

#include <cmath>

#include <algorithm>

#include <vector>

#include <iterator>

#include "math.h"

**using** **namespace** std;

**int** main()

{

**int** n;

cin >> n;

vector<**bool**> inA(3 \* n / 2 + 100, **false**);

inA[2] = **true**;

inA[3] = **true**;

inA[4] = **true**;

inA[7] = **true**;

inA[13] = **true**;

inA[15] = **true**;

**int** bim1 = 10;

**int** bim2 = 9;

**int** bim3 = 8;

**for** (**int** i = 7; i <= n; i++)

{

**int** ai = bim1 + bim3;

**if** (ai < (**int**)inA.size()) {

inA[ai] = **true**;

}

**int** bi = bim1 + 1;

**while** (inA[bi])

{

bi++;

}

bim3 = bim2;

bim2 = bim1;

bim1 = bi;

**if** (i == n)

{

cout << ai << endl;

cout << bi << endl;

**return** 0;

}

}

**int** a[] = { 0,2,3,4,7,13,15 };

**int** b[] = { 0,1,5,6,8,9,10 };

cout << a[n] << endl;

cout << b[n];

**return** 0;

}

**Приложение 5. Задача 0119. Сортировка времени**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

**struct** TIME {

**int** hours;

**int** min;

**int** sec;

};

**int** main()

{

**int** count;

cin >> count;

TIME\* A = **new** TIME[count];

**for** (**int** i = 0; i < count; i++)

{

cin >> A[i].hours >> A[i].min >> A[i].sec;

}

**for** (**int** i = 0; i<count; i++)

**for** (**int** k = 0; k < count; k++)

{

**int** timei;

**int** timek;

timei = A[i].hours \* 3600 + A[i].min \* 60 + A[i].sec;

timek = A[k].hours \* 3600 + A[k].min \* 60 + A[k].sec;

**if** (timei > timek)

{

TIME temp = A[k];

A[k] = A[i];

A[i] = temp;

}

}

**for** (**int** i = count-1; i >=0; i--)

{

cout << A[i].hours << ' ' << A[i].min << ' ' << A[i].sec << endl;

}

**return** 0;

}

**Приложение 6. Задача 0147. Числа Фибоначчи**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** f(**int** n)

{

**if** (n <= 1)

{

**if** (n == 0) **return** 0; **else** **return** 1;

}

**else** **return** f(n - 1) + f(n - 2)

;

}

**int** main()

{

**int** n;

cin >> n;

cout << f(n);

//system("pause");

**return** 0;

}

**Приложение 7. Задача 0175. Наручные часы**

#include "stdlib.h"

#include "iostream"

#include <string>

#include <vector>

**using** **namespace** std;

**int** main()

{

string str;

cin >> str;

**int** h, m;

h = stoi(str.substr(0, str.find(":")));

m = stoi(str.substr(str.find(":")+1, str.length()));

**int** dts;

**if** (h < 10)

{

dts = 20;

}

**else** **if** (h < 17)

{

dts = 24;

}

**else** **if** (h < 20)

{

dts = 24 + 5;

}

**else**

{

dts = 24 + 10;

}

cout << dts \* 60 - (h \* 60 + m);

**return** 0;

}

**Приложение 8. Задача 0203. Сдвиг текста**

#include "iostream"

#include "stdlib.h"

#include <string>

**using** **namespace** std;

**int** main()

{

**int** p, len;

**char** tmp;

string s1, s2;

getline(cin, s1);

getline(cin, s2);

len = s1.size();

p = 0;

**while** (s1 != s2)

{

tmp = s2[0];

**for** (**int** i = 0; i < len - 1; i++)

{

s2[i] = s2[i + 1];

}

s2[len - 1] = tmp;

p++;

**if** (p == len)

**break**;

}

**if** (p != len)

cout << p << "\n";

**else**

cout << -1 << "\n";

**return** 0;

}

**Приложение 9. Задача 0231. Распаковка строки**

#include <iostream>

#include <string>

#include <stdio.h>

#include <stdlib.h>

**using** **namespace** std;

**void** e(**int** c){

**if** (c % 40 == 0)

cout<<endl;

}

**int** main()

{

**freopen**("INPUT.TXT","r",stdin);

**freopen**("OUTPUT.TXT", "w",stdout);

string a, z;

**int** o = 0,c = 0;

cin>>a;

**if**(**isalpha**(a[0])){

cout<<a[0];

c++;

}

**for**(**int** i = 0; i < a.length()-1; i++){

**if**((**isalpha**(a[i])) && (**isalpha**(a[i + 1]))){

cout<<a[i+1];

c++;

e(c);

}

**else**

**if**(**isdigit**(a[i])){

z = z+a[i];

**if**(**isalpha**(a[i+1])){

o = **atoi**(z.c\_str());

**for**(**int** j = 0; j < o ; j++){

cout<<a[i+1];

c++;

e(c);

}

z.clear();

o = 0;

}

}

}

}

**Приложение 10. Задача 0315. Наименьшая система счисления**

#include <string>

#include <algorithm>

#include "iostream"

using namespace std;

int main()

{

string S;

int R = 2;

getline(cin, S);

for (int i = 0; i < S.size() && R > 0; ++i)

{

if (S[i] >= '1' && S[i] <= '9')

R = max(R, S[i] - '0' + 1);

else if (S[i] >= 'A' && S[i] <= 'Z')

R = max(R, S[i] - 'A' + 11);

else if (S[i] != '0')

R = -1;

}

cout << R << endl;

return 0;

}

**Приложение 11. Задача 0511. Очередь**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**int** t=0;

**int** col;

cin >> col;

**if** (col > 145)

cout << "NO";

**else**

**if** (col <= 2)

cout << "0 0";

**else**

{

**if** (col % 2 == 1)

t = 480 + ((col - 1) / 2) \* 10;

**else**

t = 485 + ((col - 1) / 2) \* 10;

cout << t / 60 - 8<< ' '<< t % 60;

}

**return** 0;

}

**Приложение 12. Задача 0539. Торт**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**int** N;

cin >> N;

**if** (N == 0 || N==1) cout << 0; **else**

**if** (N % 2 == 0)

{

cout << N / 2;

}

**else**

{

cout << N;

}

**return** 0;

}

**Приложение 13. Задача 0595. Слова**

#include "stdlib.h"

#include "iostream"

#include <string>

#include <vector>

**using** **namespace** std;

**int** main()

{

string scr;

string dst;

getline(cin, scr);

getline(cin, dst);

**if** (scr.size() != dst.size())

{

cout << "No";

**return** 0;

}

string s = dst + '#' + scr;

vector<**int**> prefix(1 + s.size(), 0);

**int** len = 0;

**for** (**int** i = 1; i < (**int**)s.size(); i++) {

**while** (**true**)

{

**if** (s[i] == s[len])

{

len++;

**break**;

}

**if** (len == 0) **break**;

len = prefix[len];

}

prefix[i + 1] = len;

}

**for** (**int** i = 0; i < (**int**)scr.size() - len; i++)

{

**if** (scr[i] != dst[(**int**)dst.size() - 1 - i])

{

cout << "No";

**return** 0;

}

}

cout << "Yes"<<endl;

cout << (**int**)scr.size()-len;

**return** 0;

}

**Приложение 14. Задача 0623. Снова Фибоначчи**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**long** **long** fibb(**long** **long** n) {

**return** (**pow**((1 + **sqrt**(5)) / 2, n) - **pow**((1 - **sqrt**(5)) / 2, n)) / **sqrt**(5);

}

**long** **long** last\_number(**long** **long** n) {

**return** fibb(n % 60) % 10;

}

**int** main() {

**int** num;

cin >> num;

num++;

cout << last\_number(num) << endl;;

**return** 0;

}

**Приложение 15. Задача 0651**. **Преобразование моноклеточных**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

**int** gcd(**int** a, **int** b)

{

**while** (b != 0)

{

**int** t = a%b;

a = b;

b = t;

}

**return** a;

}

**int** divCount(**int** n)

{

**if** (n == 1) **return** 0;

**int** d = 2;

**int** count = 0;

**while** (d\*d <= n)

{

**if** (n%d == 0)

{

count++;

n /= d;

}

**else**

{

d++;

}

}

**return** count + 1;

}

**int** main()

{

**int** a, b;

cin >> a >> b;

**int** g = gcd(a, b);

cout << divCount(a / g) + divCount(b / g);

**return** 0;

}

**Приложение 16. Задача 0679. Просто простые числа**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

**int** main()

{

**int** n;

cin >> n;

**int** n2, n3;

**if** (n % 3 == 0)

{

n2 = 0;

n3 = n / 3;

}

**else** **if** (n % 3 == 1)

{

n2 = 2;

n3 = (n - 4) / 3;

}

**else**

{

n2 = 1;

n3 = (n - 2) / 3;

}

**if** (n2 > 0 && n3 > 0)

{

cout << "2 " << n2 << " 3 " << n3;

}

**else** **if** (n2 > 0)

{

cout << "2 " << n2;

}

**else**

{

cout << "3 " << n3;

}

**return** 0;

}

**Приложение 17. Задача 0763. Игра с ладьей**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**long** **int** x, y;

cin >> x >> y;

**if** (x == 1 && y == 1)

{

cout << "0";

} **else**

**if** (x != y)

{

cout << "1";

} **else** cout << "2";

**return** 0;

}

**Приложение 18. Задача 0791. Соседние клетки**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**int** n;

cin >> n;

**switch** (n)

{

**case** 1:cout << "2 9"; **break**;

**case** 8:cout << "7 16"; **break**;

**case** 57:cout << "49 58"; **break**;

**case** 64:cout << "56 63"; **break**;

**case** 9: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 17: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 25: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 33: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 41: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 49: cout << n - 8 << ' ' << n + 1 << ' ' << n + 8; **break**;

**case** 16: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**case** 24: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**case** 32: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**case** 40: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**case** 48: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**case** 56: cout << n - 8 << ' ' << n - 1 << ' ' << n + 8; **break**;

**default**:

**if**(n >= 2 && n <= 7) cout << n - 1 << ' ' << n + 1 << ' ' << n + 8; **else**

**if**(n >= 58 && n <= 63) cout << n - 8 << ' ' << n - 1 << ' ' << n + 1; **else**

**if** (n<64)

cout << n - 8 << ' ' << n - 1 << ' ' << n + 1 << ' ' << n + 8;

**break**;

}

**return** 0;

}

**Приложение 19. Задача 0819.** **Прямоугольный параллелепипед**

#include "stdlib.h"

#include "iostream"

**using** **namespace** std;

**int** main()

{

**long** **long** a, b, c;

cin >> a >> b >> c;

**long** **long** result1 = (a\*b \* 2) + (b\*c \* 2) + (a\*c \* 2);

**long** **long** result2 = a\*b\*c;

cout << result1 << ' ' << result2;

//system("pause");

**return** 0;

}

**Приложение 20. Задача 0847. Анаграмма-2**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

**int** main()

{

string word1;

string word2;

cin >> word1 >> word2;

//getline(cin, word1);

//getline(cin, word2);

**if** (word1.length() == word2.length())

{

**bool** flag = **true**;

**for** (**int** i = 0; i < word1.length()&&flag; i++)

{

**if** (word1[i] == word2[i]) flag = **false**;

}

**if** (flag)

{

**do**

{

flag = **false**;

**for** (**int** i = 0; i < word1.length() - 1; i++)

{

**if** (word1[i] > word1[i + 1])

{

**char** tmp = word1[i];

word1[i] = word1[i + 1];

word1[i + 1] = tmp;

flag = **true**;

}

}

} **while** (flag);

**do**

{

flag = **false**;

**for** (**int** i = 0; i < word2.length() - 1; i++)

{

**if** (word2[i] > word2[i + 1])

{

**char** tmp = word2[i];

word2[i] = word2[i + 1];

word2[i + 1] = tmp;

flag = **true**;

}

}

} **while** (flag);

**if** (word1 == word2) cout << "YES"; **else** cout << "NO";

} **else** cout << "NO";

} **else** cout << "NO";

**return** 0;

}

**Приложение 21. Задача 0903. Бисер**

#include "stdlib.h"

#include "iostream"

#include <string>

**using** **namespace** std;

**int** main()

{

**int** n;

cin >> n;

cout << n + 1;

**return** 0;

}