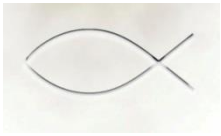
Введение

Шифрование — то есть сокрытие информации — появилось еще в древние времена. А уж когда возникли государства, армии, войны, разведка — то возникла необходимость тайно передавать какие-то сведения, чтобы, если вдруг они попадутся в руки врагу, тот ничего бы не понял. Нужны были тайные знаки, чтобы узнавать своих. Например, разрезали на части монету. Люди могли никогда друг друга не видеть, но если посланец предъявлял свою половинку, и при наложении обе части совпадали, значит, это свой.

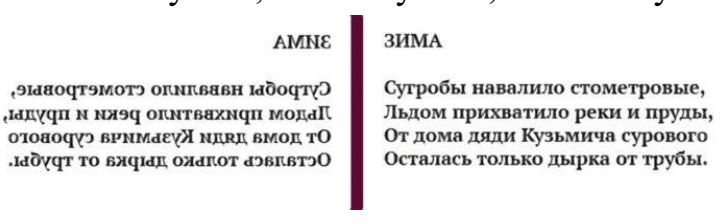
А еще такой секретный знак был у первых христиан — в те века, когда за исповедание христианской веры тебя могли казнить. Как христиане могли узнавать своих — так, чтобы никто их не заподозрил и не выдал властям? У христиан был священный знак, символическое изображение рыбы (потому что если прочитать первые буквы фразы по-гречески «Иисус Христос Божий Сын Спаситель», то получалось греческое слово «ихтис», что значило рыба). Поэтому один христианин мог начертить тростью на земле дугу — сама по себе дуга еще ничего не обозначала. Но второй христианин в ответ на это чертил другую дугу, которые вместе складывались в изображение рыбы. Вот так:



И оба понимали, что они — единоверцы. А со стороны никто бы ничего не понял.

В древности люди еще и придумывали «тайные языки», на которых можно было устно разговаривать, и никто из посторонних не мог понять эту «тарабарщину». В старину на Руси были такие люди, которые назывались *офени*. Это бродячие торговцы разным мелким товаром — гребнями, бусами, нитками, пуговицами, ленточками, иголками, ножницами и так далее. Но они не только занимались торговлей, но подчас выведывали разные тайны, то есть торговля у них служила лишь прикрытием для разведки. И вот между собой они говорили на специальном языке — брали слово и переставляли местами слоги. Если слово двусложное, то сперва говорили второй слог, потом первый. Не «палка», а «капал», не «рыба», а «бары». Если слово трехсложное, то сперва говорили третий слог, потом второй, потом первый. Вместо «рыбалка» было «кабалры». Ну и так далее.

Были и другие старинные шифры. Например, слова писались не слева направо, а справа налево. Не «капуста», а «атсупак», не «бабушка», а «акшубаб». Еще в старину часто использовали шифр, когда буквы в слове писались в зеркальном отражении. Прочитать такой текст можно было, только поднеся его к зеркалу:



Но это всё довольно простые шифры, которые очень легко разгадать. Вскоре их стали понимать многие и потребовался другой, секретный канал передачи сообщений — прежде всего, в военных целях. С тех пор люди соревнуются в изобретении самых защищенных способов шифрования информации.

Шифрование сообщение происходит для защиты содержимого. Получается, что всегда есть лица, заинтересованные в данных сведениях. Люди в любом случае добиваются успехов, находя способы расшифровки кодов. Соответственно, криптография адаптируется.

В современном виде криптография далека от банальной перестановки букв по алфавиту. Головоломки имеют невероятный уровень сложности, и их решение требует огромных вычислительных мощностей. Вместо простого смещения буквы подменяются числами, символами и проходят сотни или тысячи шагов.

С распространением компьютеров криптография выходит на новый уровень. Мощности новых устройств позволяют создавать на порядки более сложные шифры. Шифр или код становится языком общения между компьютерами, а криптография становится полноценной гражданской отраслью. В 1978 году разрабатывается стандарт шифрования DES, который стал основой для многих современных криптографических алгоритмов.

Сфера использования криптографии расширяется, при этом власти различных стран пытаются удержать контроль над использованием шифров. Разработки криптографов засекречиваются, от производителей шифровальных машин требуют оставлять в продуктах «черные ходы» для доступа спецслужб.

Параллельно независимые криптоаналитики разрабатывают способы шифрования, которыми могли бы пользоваться все желающие – так называемую открытую криптографию. Особенно актуально это стало с развитием интернета, где вопрос конфиденциальности информации встал очень остро. Первой криптосистемой с открытым ключом считается созданный в 1977 году алгоритм RSA, название которого является акронимом имен создателей – Риверста, Шамира и Адельмана. А в 1991 году американский программист Филипп Циммерман разрабатывает популярнейший пакет PGP с открытым исходным кодом для шифрования электронной почты.

Распространение доступного интернета по всему миру невозможно представить без криптографии. С появлением мессенджеров, социальных сетей, онлайн-магазинов и сайтов государственных услуг передача персональной информации в сети происходит без остановки и в огромных количествах. Сегодня мы сталкиваемся с криптографией ежедневно, когда вводим пароль от почтового сервиса, узнаем статус покупки онлайн или делаем денежный перевод через приложение банка. Криптография прошла гигантский путь от простых шифров древности к сложнейшим криптосистемам. Будущее этой науки творится на наших глазах – очередная революция в шифровании произойдет с появлением квантовых суперкомпьютеров, разработка которых уже ведется.

Требования к оформлению заданий. При проверке заданий учитывается не только ответ, но и само решение. Важно оформлять решения во всех заданиях. Ответы без решения оцениваются не более чем в 1 балл.

Задача 1 *(Максимум 10 баллов)* **Сэр Френсис Бэкон**

Шифр Бэкона — метод сокрытия секретного сообщения, придуманный Фрэнсисом Бэконом в начале XVII века. Он разработал шифр, который бы позволил передавать секретные сообщения в обычных текстах так, чтобы никто не знал об этих сообщениях.

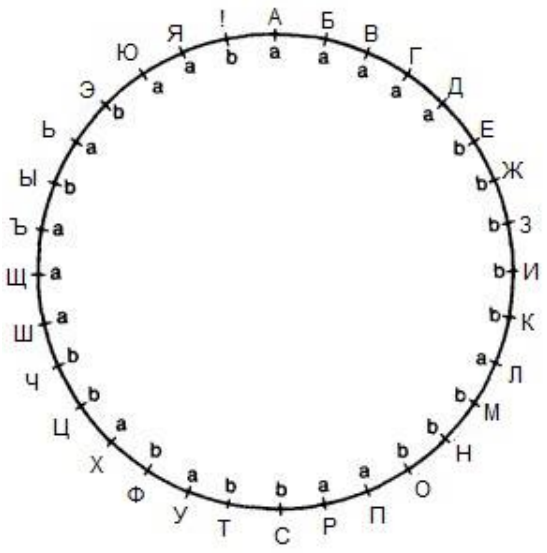
Суть его шифра:

1) замена букв алфавита на последовательность из пяти символов a и b согласно ключу. Брались 5 символов, отсчитывая с этой буквы, например, И=*bbabb*.

2) Выбиралось правило сокрытия в другом тексте, например, a-заглавная, b-строчная.

3) Текст видоизменялся или подбирался для соответствия правилу. Например, текст «МАгИЯ» по правилу из 2 пункта соответствует символам *aabaa*, т.е. Ю.

Ключ этого шифра:



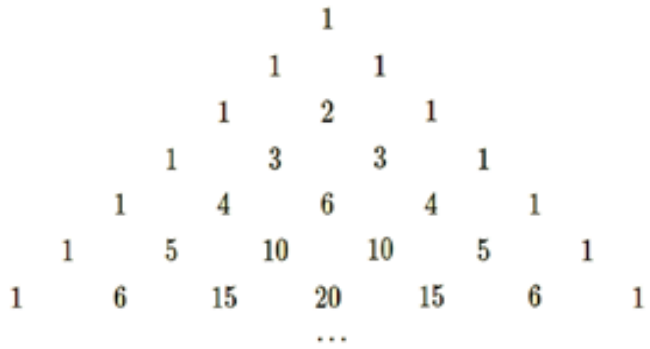
Что скрыто в этом сообщении:

С дНеМ рождеНИя, ЛЮБИмыЙ сынок!

Задача 2 *(Максимум 20 баллов)* **Треугольник**

Треугольник Паскаля - список чисел, выстроенных виде некоторого треугольника. Этот список можно продолжать постоянно. В первой строке стоит одна цифра «1». Во второй стоят две цифры «1». Начиная с третьей по бокам стоят «1», а внутренние числа заполняются при сложении тех чисел, которые стоят непосредственно над ними.

Сколько четных чисел содержится в строке с номером 110?



Задача 3 *(Максимум 20 баллов)* **Проверка**

Некое устройство генерирует пароли в виде двоичной последовательности по следующей схеме:

1) Генерируется случайные 2 байта X (каждый бит случаен);
2) Из двоичной записи побитово генерируются новые 2 байта Y по следующему правилу: $y_i=x_i*x_{i+1}$, $y_{15}=0$, $i\in[0,14]$; x_i,y_i – биты.
3) 2 байта Y представляют собой сгенерированный пароль.

Какие из представленных паролей может сгенерировать устройство:

а) 1110110000100111

б) 1111000001100110

в) 0001110111001l0

г) 0000111001001l0

Задача 4 *(Максимум 20 баллов)* **Поиск**

Вася поделился с другом Вовой паролем к своей домашней Wi-Fi сети. Вова ради смеха захотел узнать названия всех социальных сетей, которыми пользуется Вася. Чтобы перехватить эти сведения, он использовал специальную программу (сниффер). Вова не хочет проверять все собранные данные вручную, поэтому он использовал функцию сниффера по анализу содержимого html-страниц с применением регулярных выражений. Помогите Вове написать регулярное выражение, позволяющее выделить искомые названия.

Код типовой страницы авторизации имеет следующий вид:

```
<html>
<head>
<title>Социальная сеть «телеграфф»</title>
 </head>
<body>
 <form>
<input type="hidden" name="login" value="vasiliymaster@gmail.com"/>
<input type="text" id="23562234" name="login"
value=«vasiliymaster@gmail.com» class="loginstring"/>
 <input type="hidden" name="password" value="mycodeisthebest"/>
<input type="password" name="password" class="passwordstring"/>
<input type="submit" name="action" value="1 онтөернить" />
</form>
</body>
</html>
```

Комментарий

Регулярные выражения предоставляют возможности для описания подстрок определенного вида. Для формирования регулярного выражения используются следующие элементы:

1. символ – например, «a»

2. любой символ – обозначается «.»

3. пробельный символ – обозначается «\s»

4. диапазон символов – обозначается «[]». Например:

[abc] – любая из букв a, b, c

[a-z0-9] – любая из малых букв латинского алфавита и цифра

5. отрицание диапазона:

[^a-z5] – не маленькая буква латинского алфавита и цифра 5

Для указания количества вхождений используются кванторы:

- «?» означает «0 или 1шт.»
- «+» означает «>= 1шт.», причем берется как можно большее количество символов
- «+?» означает «>= 1шт.», причем берется как можно меньшее количество символов
- «*» означает «>= 0шт.», причем берется как можно большее количество символов
- «*?» означает «>= 0шт.», причем берется как можно меньшее количество символов
- «{5}» означает «5 шт.»
- «{5, 8}» означает «от 5 до 8 шт.»

Например, «a*» – любое количество идущих подряд символов «a».

Для извлечения подстроки используются круглые скобки, например, выражение: ([0-9]+)\s+([0-9]+) сохраняет из строки «мои числа 45 567 234 56» числа: «45» и «567».

Практическое задание. *(Максимум 30 баллов)*

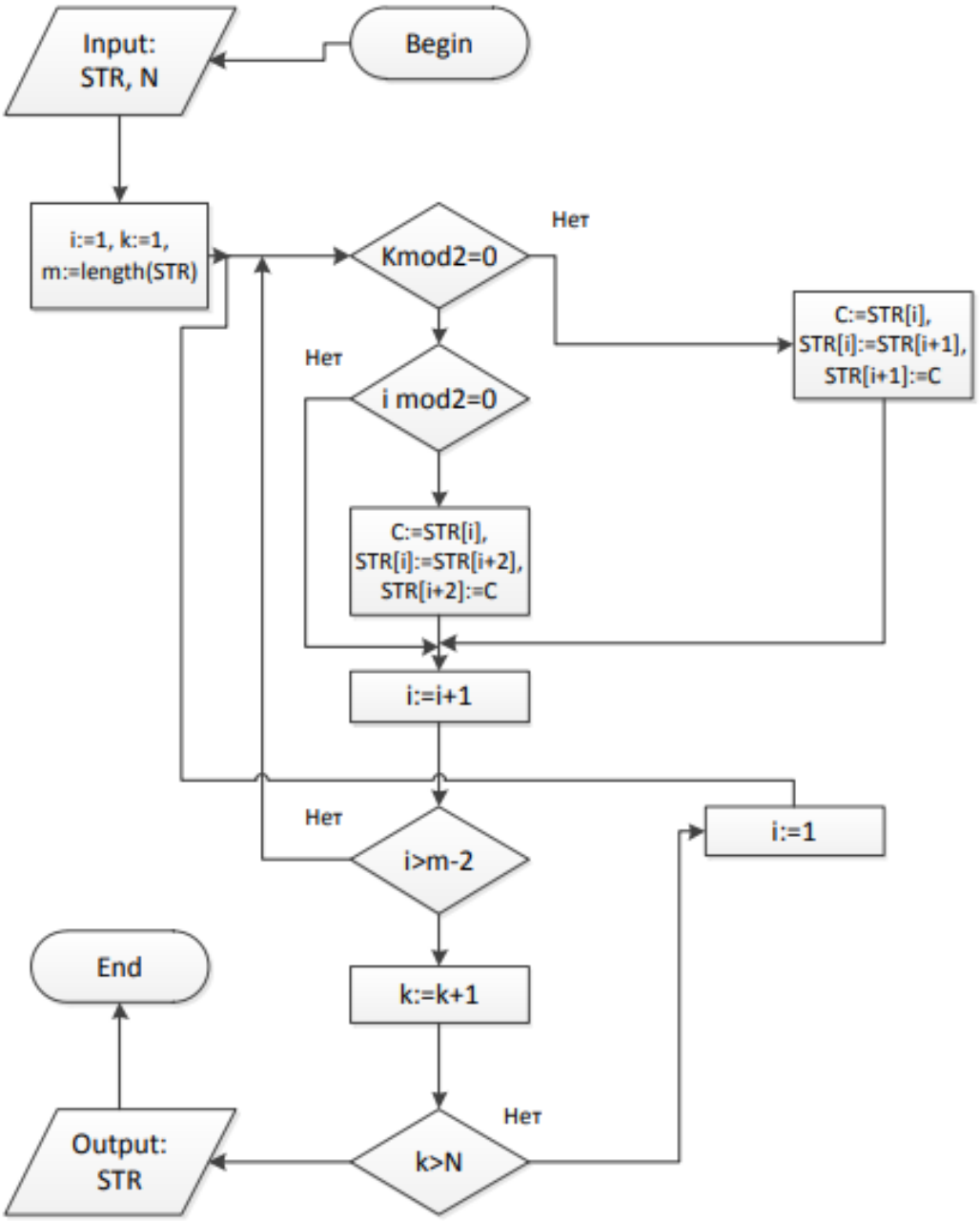
Скремблирование — это обратимое преобразование цифрового потока без изменения скорости передачи с целью получения свойств, близких к свойствам случайной последовательности. Исходное сообщение можно восстановить, применив обратный алгоритм. Применительно к телекоммуникационным системам скремблирование повышает надежность синхронизации устройств, подключенных к линии связи, и уменьшает уровень помех, излучаемых на соседние линии многожильного кабеля. Есть и иная область применения скремблеров — защита передаваемой информации от несанкционированного доступа.

Методы скремблирования бывают разные. Например, один из самых простых – разбиение исходного сообщения на пары символов и перемена символов каждой паре местами. Например, слово ЯБЛОКО после такого алгоритма превратится в БЯОЛОК.

Ниже представлен некий алгоритм скремблирования в виде блок-схемы.

Задания:

1) Опишите принцип действия алгоритма;
2) Напишите, что будет на выходе, если на основной алгоритм подать N=5 и STR= «password».


Begin
Input:
STR, N
i:=1, k:=1,
m:=length(STR)
Kmod2=0
Нет
C:=STR[i],
STR[i]:=STR[i+1],
STR[i+1]:=C
Нет
i mod2=0
C:=STR[i],
STR[i]:=STR[i+2],
STR[i+2]:=C
i:=i+1
Нет
i>m-2
i:=1
End
k:=k+1
Нет
k>N
Output:
STR