

Пояснения к олимпиадным заданиям
Информационная безопасность
10-11 класс

Алексеевский П.И.

Пояснение к заданиям 12-13



- В этих заданиях следует обратить внимание на права доступа к файлам. В большинстве современных операционных систем (различные варианты Linux, семейство BSD, Solaris, macOS, Android, iOS и др.) права доступа задаются одинаково.

Пояснение к заданиям 12-13



- Подробный вывод содержимого директории включает три нужных нам поля — режим доступа (задаётся командой `chmod`), владелец (задаётся командой `chown`) и группа (команды `chown` и `chgrp`).
- Например:
 - ```
-rw-r--r-- 1 test users 15 авг 17 13:28 hello.txt
```

|     |          |       |        |
|-----|----------|-------|--------|
|     | _____    | _____ | _____  |
|     | Режим    |       | Группа |
| Тип | Владелец |       |        |

# Пояснение к заданиям 12-13



- Наибольший интерес для нас представляет режим. Он состоит из трёх элементов (разрешённых действий) для трёх субъектов — владельца, группы и всех остальных. Если соответствующее разрешение отсутствует, то в этом месте в списке файлов указывается знак «-».

|   | Владелец |        |                     | Группа |        |                     | Остальные |        |                     |
|---|----------|--------|---------------------|--------|--------|---------------------|-----------|--------|---------------------|
|   | Чтение   | Запись | Исполнение или вход | Чтение | Запись | Исполнение или вход | Чтение    | Запись | Исполнение или вход |
| - | r        | w      | x                   | r      | w      | x                   | r         | w      | x                   |

# Пояснение к заданиям 12-13



- В приведённом ранее примере:
- `-rw-r--r-- 1 test users 15 авг 17 13:28 hello.txt`
- видно, что право на исполнение файла отсутствует у всех субъектов, право на чтение есть у всех, а право на запись есть только у владельца.

|   | Владелец |        |                     | Группа |        |                     | Остальные |        |                     |
|---|----------|--------|---------------------|--------|--------|---------------------|-----------|--------|---------------------|
|   | Чтение   | Запись | Исполнение или вход | Чтение | Запись | Исполнение или вход | Чтение    | Запись | Исполнение или вход |
| - | r        | w      | -                   | r      | -      | -                   | r         | -      | -                   |

# Пояснение к заданиям 12-13



- Для экономии времени и места часто эти разрешения записываются в виде числа в **восьмеричной** системе счисления. Для понимания принципа, можно записать разрешения в виде **двоичного** числа, где 1 обозначает наличие разрешения, а 0 — отсутствие. В приведённом примере это будет число  $110100100_2$ . Если перевести его в восьмеричную систему счисления, получится короткое и удобное в использовании значение  $644_8$ . В командах, управляющих разрешениями (например, `chmod`) основание системы счисления не указывают, поэтому использоваться будет только значение числа **644**.

|   | Владелец |        |                     | Группа |        |                     | Остальные |        |                     |
|---|----------|--------|---------------------|--------|--------|---------------------|-----------|--------|---------------------|
|   | Чтение   | Запись | Исполнение или вход | Чтение | Запись | Исполнение или вход | Чтение    | Запись | Исполнение или вход |
| - | r        | w      | -                   | r      | -      | -                   | r         | -      | -                   |

# Пояснение к заданиям 11 и 20



- Для решения этих заданий потребуются умение понимать алгоритмы, записанные на каком-либо языке программирования. Примеры программного кода приведены на двух часто используемых языках — C++ и Pascal.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- В данном примере представлена функция, принимающая на вход строку и возвращающая результат также в виде строки. Из названия функции можно предположить, что она каким-то образом осуществляет шифрование. Поскольку больше никаких данных в неё не передаётся, шифрование осуществляется либо без ключа, либо с фиксированным ключом.



# Пояснение к заданиям 11 и 20

```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Рассмотрим, что именно происходит в этой функции. Основная часть её — это цикл, перебирающий все символы исходной строки. Переменная `ch` внутри цикла принимает значение каждого из символов исходной строки.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Поскольку не все диалекты языка Pascal поддерживают арифметические операции с символьными переменными, в этом примере используются ASCII-коды соответствующих символов. Для конвертации символа в его код используется функция **ord**, для обратного преобразования — **chr**. В языке C++ арифметические операции над символами работают независимо от версии стандарта, поэтому дополнительной конвертации не требуется.
- Итак, главное, что нас будет интересовать — функция выполняет манипуляции с кодами символов.

# Пояснение к заданиям 11 и 20

```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Также внутри цикла используется целочисленная переменная  $i$ . Судя по алгоритму, эта переменная принимает значения 1, 2, 3 и т. д., т. е. это порядковый номер символа в исходной строке, считая от 1.

# Пояснение к заданиям 11 и 20

```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Код символа подвергается преобразованию только в том случае, если этот символ — не пробел. Преобразование заключается в том, чтобы прибавить к коду символа порядковый номер его в исходной строке.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Если после преобразования код символа превышает таковой для буквы «z», то код уменьшается на количество букв в латинском алфавите. Таким образом, после буквы «z» снова следует буква «a».
- Можно также заметить, что функция будет корректно работать только со строчными латинскими буквами.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
std::string crypt(const std::string_view& msg)
{
 int i = 1;
 std::string result;
 for(char ch: msg) {
 if (ch != ' ') ch = ch + i;
 i = i + 1;
 while (ch > 'z') ch = ch - 26;
 result = result + ch;
 }
 return result;
}
```

```
(* Пример кода на языке Pascal *)
function crypt(var msg: string): string;
var i, k, c: integer; result: string; ch: char;
begin
 i := 1; result := "";
 for k := 1 to length(msg) do begin
 ch := msg[k]; c := ord(ch);
 if ch <> ' ' then c := c + i;
 i := i + 1;
 while c > ord('z') do c := c - 26;
 result := result + chr(c);
 end;
 exit(result);
end;
```

- Таким образом, полностью алгоритм можно понять так:
- **Каждый символ из алфавита «a...z» циклически сдвигается вперёд на величину, являющуюся порядковым номером этого символа в строке.**

# Пояснение к заданиям 11 и 20

- Например, попробуем зашифровать по этому алгоритму строку «hello world»:

|       |   |   |   |   |   |   |   |   |   |    |    |
|-------|---|---|---|---|---|---|---|---|---|----|----|
| №     | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| До    | h | e | l | l | o |   | w | o | r | l  | d  |
| После | i | g | o | p | t |   | d | w | a | v  | o  |

**Каждый символ из алфавита «a...z» циклически сдвигается вперёд на величину, являющуюся порядковым номером этого символа в строке.**

# Пояснение к заданиям 11 и 20

- Соответственно, для расшифрования потребуется сдвинуть символы в обратную сторону:

|       |   |   |   |   |   |   |   |   |   |    |    |    |    |
|-------|---|---|---|---|---|---|---|---|---|----|----|----|----|
| №     | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| До    | i | c | y | i |   | g |   | k | x | y  | v  | u  | r  |
| После | h | a | v | e |   | a |   | c | o | o  | k  | i  | e  |

**Каждый символ из алфавита «a...z» циклически сдвигается вперёд на величину, являющуюся порядковым номером этого символа в строке.**



# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
bool check(const std::string_view& pw)
{
 int l = pw.length();
 int s = 0;
 if (l < 2) return false;
 for(char ch: pw) {
 if (ch < 'a' or ch > 'z') return false;
 s = s + (ch - 'a') + 1;
 }
 if (s % l == 3) return true;
 return false;
}
```

```
(* Пример кода на языке Pascal *)
function check(var pw: string): boolean;
var l, s, i: integer; ch: char;
begin
 l := length(pw); s := 0;
 if l < 2 then exit(false)
 else
 for i := 1 to l do begin
 ch := pw[i];
 if (ch < 'a') or (ch > 'z') then exit(false);
 s := s + (ord(ch)-ord('a')) + 1;
 end;
 if s mod l = 3 then exit(true);
 exit(false);
end;
```

- В этом примере представлена функция, принимающая на вход строку и возвращающая булево значение (истина или ложь). Из названия функции можно предположить, что она что-то проверяет. По условию задачи это проверка «правильности» пароля.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
bool check(const std::string_view& pw)
{
 int l = pw.length();
 int s = 0;
 if (l < 2) return false;
 for(char ch: pw) {
 if (ch < 'a' or ch > 'z') return false;
 s = s + (ch - 'a') + 1;
 }
 if (s % l == 3) return true;
 return false;
}
```

```
(* Пример кода на языке Pascal *)
function check(var pw: string): boolean;
var l, s, i: integer; ch: char;
begin
 l := length(pw); s := 0;
 if l < 2 then exit(false)
 else
 for i := 1 to l do begin
 ch := pw[i];
 if (ch < 'a') or (ch > 'z') then exit(false);
 s := s + (ord(ch)-ord('a')) + 1;
 end;
 if s mod l = 3 then exit(true);
 exit(false);
end;
```

- В основе приведённого алгоритма лежит, в очень упрощённом виде, некая хэш-функция, которая преобразует строку произвольной длины в некоторое число. В данном случае вычисляется сумма порядковых номеров символов исходной строки в алфавите «а...z» и находится остаток от деления этой суммы на длину строки.

# Пояснение к заданиям 11 и 20



```
/* Пример кода на языке C++ */
bool check(const std::string_view& pw)
{
 int l = pw.length();
 int s = 0;
 if (l < 2) return false;
 for(char ch: pw) {
 if (ch < 'a' or ch > 'z') return false;
 s = s + (ch - 'a') + 1;
 }
 if (s % l == 3) return true;
 return false;
}
```

```
(* Пример кода на языке Pascal *)
function check(var pw: string): boolean;
var l, s, i: integer; ch: char;
begin
 l := length(pw); s := 0;
 if l < 2 then exit(false)
 else
 for i := 1 to l do begin
 ch := pw[i];
 if (ch < 'a') or (ch > 'z') then exit(false);
 s := s + (ord(ch)-ord('a')) + 1;
 end;
 if s mod l = 3 then exit(true);
 exit(false);
end;
```

- Проверка считается пройденной успешно, если длина пароля — 2 символа или больше, и вычисленный хэш равен 3. При решении данной задачи следует обратить внимание на то, что не для любой длины пароля существуют такие значения хэша. Для строки длиной  $N$  максимальным значением остатка от деления на эту длину будет  $N-1$ , таким образом, несмотря на проверку длины в начале функции, в данном примере пароль должен содержать по крайней мере 4 символа.

Ответы на остальные задания см. в ключе.