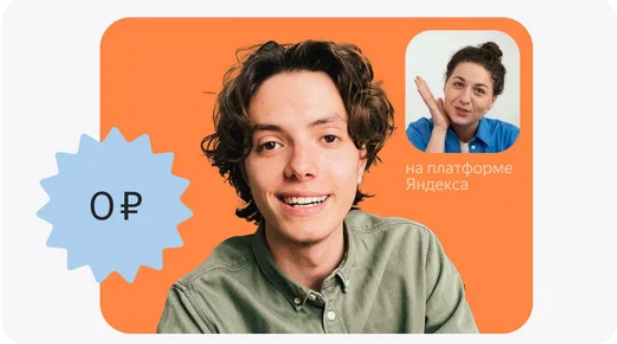


ХОЧУ ПОМОЧЬ
ПРОЕКТУ

Модуль crypt в Python, хеширование паролей



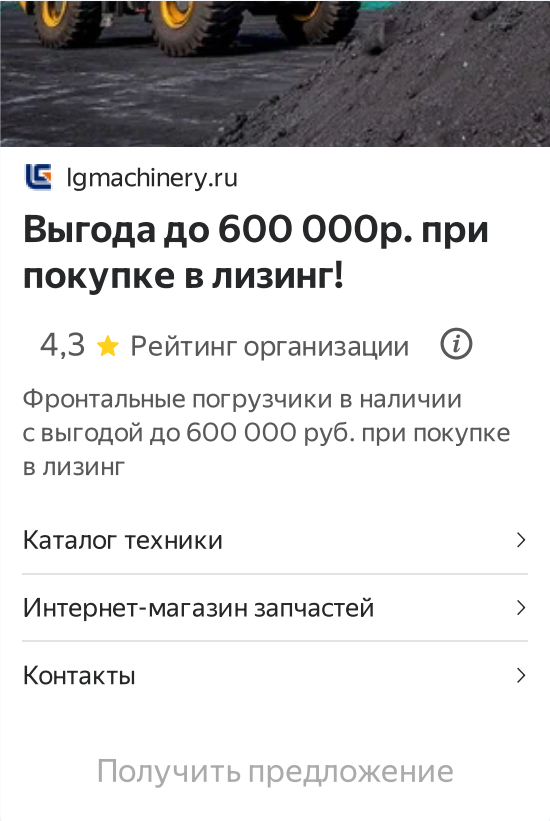
practicum.yandex.ru

РЕКЛАМА · 18+ Я

Бесплатное занятие английским в Яндекс Практикуме

Полноценное занятие с преподавателем, а не презентация курсов

Узнать больше



Igmachinery.ru

Выгода до 600 000р. при покупке в лизинг!

4,3 ★ Рейтинг организации ⓘ

Фронтальные погрузчики в наличии с выгодой до 600 000 руб. при покупке в лизинг

Каталог техники

Интернет-магазин запчастей

Контакты

Получить предложение

Модуль crypt в Python, хеширование паролей

Удаление и проверки хешированных паролей

Удаление паролей на основе функции `crypt(3)`, которая является односторонней хеш-функцией, алгоритме DES. Возможное использование включает хранение хешированных паролей, чтобы охраняя действительный пароль или пытаться взломать пароли Unix с помощью словаря.

удален из стандартной библиотеки с версии Python 3.13. С версии Python 3.11 будет устаревании.

не реализует хеширование паролей, в основном, старыми, некачественными и пользователям не рекомендуется их использовать.

*cross-platform. Кроссплатформенные приложения в любом случае нуждаются в альтернативной реализации DES. DES имеет крайне ограниченное пространство ключей 2**56.*

- MD5, SHA256 с солью, SHA512 с солью и Blowfish являются необязательными расширениями. SSHA256 и SSHA512 являются расширениями glibc. Blowfish (bcrypt) - единственный безопасный алгоритм. Однако он находится в glibc и поэтому обычно не доступен в Linux.*
- В зависимости от платформы, модуль crypt не является потокобезопасным. Только реализации с crypt_r(3) являются потокобезопасными.*
- Модуль никогда не был полезен для взаимодействия с системными базами данных пользователей и паролей. В BSD, macOS и Linux все операции аутентификации пользователя и изменения пароля должны проходить через PAM (подключаемый модуль аутентификации).*

Чем можно заменить: модулем стандартной библиотеки [hashlib](#).

Обратите внимание, что поведение модуля crypt зависит от фактической реализации подпрограммы crypt(3) в работающей системе. Поэтому любые расширения, доступные в текущей реализации, также будут доступны в этом модуле.

Доступность: Unix. Недоступно на VxWorks и Windows.

Примеры:

Простой пример, иллюстрирующий типичное использование. Операция сравнения с постоянным временем необходима для ограничения подверженности атакам времени. Для этой цели подходит `hmac.compare_digest()`:

```
import pwd
import crypt
import getpass
from hmac import compare_digest as compare_hash

def login():
    username = input('Python login: ')
    cryptpasswd = pwd.getpwnam(username)[1]
    if cryptpasswd:
        if cryptpasswd == 'x' or cryptpasswd == '*':
            raise ValueError('no support for shadow passwords')
        cleartext = getpass.getpass()
        return compare_hash(crypt.crypt(cleartext, cryptpasswd), cryptpasswd)
```

Вверх


https://docs-python.ru/standart-library/modul-crypt-python/

1/2

```
else:
    return True
```

Сгенерировать хеш пароль, используя самый надежный из доступных методов и сравнить его с оригиналом:

РЕКЛАМА



lgmachinery.ru

Выгода до 600 000р. при покупке в лизинг!

4,3 ★ Рейтинг организации ⓘ

Фронтальные погрузчики в наличии с выгодой до 600 000 руб. при покупке [DOCS-Python.ru™](#), 2023 г.

Каталог техники >

Интернет-магазин запчастей >

Контакты >

Получить предложение

```
def compare_hash(password, hashed):
    try:
        crypt.crypt(password, hashed):
        if password != hashed:
            return False
        return True
    except:
        return False
```

Содержание раздела:

[Модуль crypt](#)

(Внимание! При копировании материала ссылка на источник обязательна)

[@docs_python_ru](#)