(ɔ •◡•)ɔ ♥ Поступашки - ШАД, Стажировки и Магистратура ♥

t.me/postypashki-old

Введение-содержание

Помимо благотворительной деятельности, Поступашки также проводят курсы и индивидуальные занятия по подготовке к ШАД, к олимпиадам, к собеседованиям, подготовке абитуриентов к ВУЗовской программе, подготовке к экзаменам, контрольным и прочим студентческим работам по основным математическим диссциалинам: анализ, линейная алгебра, теория вероятностей, теория групп и тд. А также по алгоритмам и структурам данных, ML&DL.

Подробнее об индивидуальных занятиях

Подробнее о курсах

A. Так называемое сжатие

Задача написать функцию `getCompressedString (text: string): string`, которая на вход принимает строку с текстом.

Функция должна вернуть строку, в которой все слова заменены на индексы слов текста из массива, в котором эти слова должны быть отсортированы в порядке частоты встречаемости в тексте (если частота встречаемости нескольких слов совпадает, то сортируем их в таком порядке, в каком они встречаются в тексте).

Знаки пунктуации, пробелы и переносы строк нужно оставить как есть. Для простоты в тексте будут встречаться следующие знаки препинания — точка, запятая, восклицательный и вопросительный знаки.

Слова с одними и теми же буквами, но разного регистра, будем считать одинаковыми. То есть «олег» и «Олег» — одно и то же слово.

Ограничения

- длина файла не должна превышать 10000 символов
- решение будет запускаться на nodejs 20.14.0 с `"type": "module"` в `package.json` (соответственно нужно делать `export function getCompressedString`)
- в тексте программы запрещено использование следующих слов: `import`, `require`, `eval`, `global`, `globalThis`, `process`, `module`

Примеры

Пример №1

Ввод

```
Hello my name is Vitaliy! And what is your name?
```

Вывод

```
2 3 0 1 4! 5 6 1 7 0?
```

Пример №2

Ввод

```
Привет, как у тебя дела?
Да, вроде, хорошо, а у тебя?
```

Вывод

```
2, 3, 0, 1, 4)
5, 6, 7, 8 0 1?
```


Шаблон решения

```
export function getCompressedString(text) {
    // ваш код
}
```

B. Проверка последовательностей

На вход подается список правил сортировки и список последовательностей чисел. Правило сортировки имеет вид X|Y и обозначает, что если в последовательности встречаются оба числа X и Y, то X обязательно должен входить в последовательность раньше, чем Y. Последовательность чисел представлена массивом чисел, все числа последовательности разные. Необходимо определить количество корректных последовательностей для заданных правил сортировки.

Формат ввода

Входные данные представлены одной строкой со следующей структурой

- Сначала перечисляются правила в формате X|Y, разделенные символом переноса строки.
- Затем следует пустая строка.
- Затем перечисляются последовательности чисел, разделенные символом переноса строки. Числа в последовательности разделены пробелом и запятой.

Примечания

Например, рассмотрим входные данные:

```
28|12
5|9
9|2


10, 5, 15, 28, 9, 12
2, 7, 12, 5, 9
4, 8, 16, 32, 64
```

В этом примере:

- Первая последовательность корректна, так как соблюдаются правила `28|12` и `5|9`, а правило `9|2` неприменимо.
- Вторая последовательность некорректна, так как нарушено правило `9|2`.
- Третья последовательность корректна, потому что ни одно из правил неприменимо.

Итак, в данном списке 2 корректные последовательности.

Ваше решение должно быть оформлено в виде функции, которая принимает единственный аргумент — строку со структурой, описанной выше. Функция должна возвращать число — количество корректных последовательностей.

Шаблон решения

```
function solution(input) {
    // ваше решение
}

module.exports = solution;
```

C. Извлечение записей

Легенда

Искусственный интеллект, который вы разрабатывали для хранения и обработки научных данных, вышел из-под контроля. Перед блокировкой доступа к своей базе знаний он активировал защиту, сделав процесс получения записей крайне сложным. Вам необходимо найти способ обойти ограничение, чтобы извлечь критически важные данные, которые помогут понять причину сбоя и восстановить контроль над ИИ.

Вам поручена ответственная задача — разработать функцию, которая поможет получить необходимые научные записи из поврежденной базы данных, управляемой ИИ.

Условие

Напишите асинхронную функцию, которая принимает `url` и `recordId` а возвращает `Promise` с объектом который сдержит три метода:

- `getTitle`: метод возвращает заголовок записи (поле `title`).
- `getSummary`: метод возвращает краткое описание (поле `summary`).
- `getDetails`: метод возвращает подробное описание (поле `details`).

Данные базы знаний необходимо отфильтровать, чтобы получить конкретную запись по указанному идентификатору `recordId`. Сервер отвечает в формате JSON со следующей структурой:

```
{
  "records": [
    {
      "id": 101,
      "title": "Теория относительности",
      "summary": "Теория относительности в кратком описании.",
      "details": "Теория относительности Альберта Эйнштейна объясняет природу гра
    },
    {
      "id": 102,
      "title": "Квантовая механика",
      "summary": "Введение в квантовую механику.",
      "details": "Квантовая механика описывает физику на малых масштабах, включая
    },
    ...
  ]
}
```

```
нии.",
йна объясняет природу гравитации как результат искривления пространства-времени."


 малых масштабах, включая поведение атомов и субатомных частиц."
```


Обработка ошибок

Если запись с указанным `id` не найдена, выбросьте ошибку в формате:

```
new Error(`Запись не найдена, id: ${recordId}`);
```

Если API вернул данные в неожиданном формате (вместо ожидаемой структуры JSON), выбросьте ошибку в формате:

```
new Error(`Неожиданный формат данных: ${url}`);
```

Шаблон решения

```
function getRecord(url, recordId) {
    // Your code here...
}

module.exports = getRecord;
```

D. Создатель встреч

В вашем корпоративном календаре сломался планнер встреч, который отвечал за создание встреч и проверку занятости участников.

Вам необходимо реализовать функцию `createMeeting`, которая принимает на вход конфиг, который содержит список встреч сотрудников и параметры создаваемой встречи, а результатом должна вернуть статус создания встречи.

Шаблон решения

```
module.exports = function createMeeting(config) {
    // ваш код
}
```

Формат ввода

Входные данные представляют из себя конфигурационный объект `config`, который состоит из 2 полей:

1. `meetings` — массив встреч сотрудников в формате:

```
[
  {
    from: number,
    to: number,
    person: string,
  }
]
```

где `from` и `to` — времена начала и окончания встречи, в диапазоне от 0 до 23 включительно, а `person` — уникальное имя сотрудника.

2. `params` — параметры встречи, которую нужно создать в формате:

```
{
  from: number,
  to: number,
  persons: string[],
}
```

где `from` и `to` — времена начала и окончания встречи, в диапазоне от 0 до 23 включительно, а `persons` — массив уникальных имён сотрудников.

Формат вывода

Вернуть ответ в формате:

```
{
  status: 'CREATED' | 'REJECTED',
  reason: string[] | null,
}
```

где `status` — поле, которое представляет статус ответа (`CREATED` | `REJECTED`), `reason` — поле, которое представляет собой массив имен сотрудников отсортированных в алфавитном порядке, чьи встречи накладываются.

Пример №1

Ввод

```
{
  "meetings": [
    {
      "from": 1,
      "to": 3,
      "person": "Петя"
    },
    {
      "from": 2,
      "to": 4,
      "person": "Вася"
    },
    {
      "from": 4,
      "to": 6,
      "person": "Костя"
    },
    {
      "from": 2,
      "to": 4,
      "person": "Женя"
    }
  ],
  "params": {
    "from": 3,
    "to": 4,
    "persons": [
      "Петя",
      "Костя"
    ]
  }
}
```

Вывод

```
{
  status: 'CREATED'
}
```

Пример №2

Ввод

```
{
  "meetings": [
    {
      "from": 1,
      "to": 3,
      "person": "Петя"
    },
    {
      "from": 2,
      "to": 4,
      "person": "Вася"
    },
    {
      "from": 4,
      "to": 6,
      "person": "Костя"
    },
    {
      "from": 2,
      "to": 4,
      "person": "Женя"
    }
  ],
  "params": {
    "from": 3,
    "to": 5,
    "persons": [
      "Петя",
      "Костя",
      "Женя"
    ]
  }
}
```

Вывод

```
{
  status: 'REJECTED',
  reason: ['Женя', 'Костя']
}
```

Примечания

- Для статуса `CREATED` в `reason` можно возвращать `null` или `[]`, а также можно не возвращать это поле.
- Все встречи идут минимум час.
- Если у сотрудника нет встреч, то он свободен весь день.

E. Список с подсветкой поиска

Стажёру Алёше нужно доработать существующую страничку со списком базовых HTML-тегов. Страница полностью свёрстана и менять её нельзя, но нужно сделать, чтобы поиск «заработал»: при сабмите формы поиска список должен отфильтровываться, а совпадающие подстроки должны подсвечиваться. Чтобы поиск был максимально user-friendly, не нужно учитывать регистр букв и пробелы слева и справа от символов, но нужно искать по всем символам, которые встречаются в элементах списка. Не забудьте обработать пустое значение строки поиска.

Для разработки и тестирования скачайте шаблон и макет. В качестве решения принимается файл `index.js`.

Примечания

1. Весь HTML статичный, включая элементы списка, и изменяться в тестах не будет. Для локального тестирования вам нужно добавить ссылку на своё решение внизу страницы.
2. Тестирование будет производиться скриншотами, поэтому важно не менять стили.
3. Для подсветки может использоваться любой способ, если на экране он выглядит как на макете.
4. Для подсветки используется стандартный цвет `yellow`.