Семинар #1

Антон Кухтичев

Содержание занятия

- Пароли (easy)
- Общий префикс (easy)
- Валидные скобки I (easy)
- Валидные скобки II (easy)
- Слияние двух односвязных списков (easy)
- Валидная анаграмма (easy)
- Функция split (easy)
- Пути бинарного дерева (easy)
- Проверить, что бинарное дерево это бинарное дерево поиска (medium)
- Найти все анаграммы в строке (medium)

Пароли (easy)

Пароль от некоторого сервиса должен удовлетворять таким ограничениям:

- состоять из символов таблицы ASCII с кодами от 33 до 126;
- быть не короче 8 символов и не длиннее 14;
- из 4 классов символов — большие буквы, маленькие буквы, цифры, прочие символы — в пароле должны присутствовать не менее трёх любых.
- Напишите программу, которая проверит, что введённый пароль подходит под эти ограничения.

Ссылка:

https://interview.cups.online/live-coding/?room=29637c66-0524-4578-921c-7d3ac1b11e7f

Общий префикс (easy)

Напишите функцию для вычисления наибольшего общего префикса строк, переданных в векторе words:

```
std::string common_prefix(const std::vector<std::string>& words);
```

Например, для пустого вектора функция должна вернуть пустую строку, а для вектора из строк "apple", "apricot" и "application" — строку "ap".

Ссылка:

https://interview.cups.online/live-coding/?room=b9fb7659-4490-4520-b734-a8e3f5af0599

Валидные скобки I (easy)

Напишите функцию для проверки валидности скобок ‘(’, ‘)’:

- Открывающая скобочка должна быть закрыта;
- Открывающая скобочка должна быть закрыта в правильном порядке;
- У каждой закрывающей скобочки есть соответствующая открывающая скобочка.

Ссылка:

https://interview.cups.online/live-coding/?room=42dc1417-c704-4d01-b315-a161745e2e21

Валидные скобки II (easy)

Напишите функцию для проверки валидности скобок `‘(’, ‘{’, ‘[’, ‘]’, ‘}’ ‘)’`:

- Открывающая скобочка должна быть закрыта;
- Открывающая скобочка должна быть закрыта в правильном порядке;
- У каждой закрывающей скобочки есть соответствующая открывающая скобочка.

Ссылка:

https://interview.cups.online/live-coding/?room=42dc1417-c704-4d01-b315-a161745e2e21

Слияние двух односвязных списков (easy)

- Есть два односвязных списка, отсортированных по возрастанию
- Нужно слить эти два списка в один

```
std::forward_list<int32_t> mergeTwoLists(std::forward_list<int32_t>
list1, std::forward_list<int32_t> list2)
```

Ссылка:

https://interview.cups.online/live-coding/?room=f8a2fb72-9f93-43cf-962c-59ff7b2ecf5b

Валидная анаграмма (easy)

- Даны две строки
- Проверить, что строка s1 это анаграмма строки s2.

```
bool is_anagram(const std::string & s1, const std::string &s2);
```

Ссылка:

https://interview.cups.online/live-coding/?room=28c811fb-1d78-4d32-b3c3-9422190f154d

split (easy)

- Есть строка str
- Есть разделитель delimiter
- Нужно разбить строку str по delimiter и вернуть std::vector подстрок

```
void split(const std::string & str, char delimiter,
std::vector<std::string> &result);
```

Ссылка:

https://interview.cups.online/live-coding/?room=507f0005-95a9-4f71-bed7-e8257ad53bd0

Пути бинарного дерева (easy)

- Дано бинарное дерево
- Нужно найти все пути от-корня-до-листа в любом порядке
- Лист - это узел без дочерних узлов

```
std::vector<std::string> binaryTreePaths(TreeNode* root);
```

Ссылка:

https://interview.cups.online/live-coding/?room=5d7628bd-da8a-4f97-bb46-27e1232aa36a

Проверить, что бинарное дерево это бинарное дерево поиска (medium)

- Дано бинарное дерево
- Нужно, что бинарное дерево это бинарное дерево поиска

```
bool isValidBST(TreeNode* root)
```

Ссылка:

https://interview.cups.online/live-coding/?room=df0d76e0-068a-452d-a34f-e9cbaa8ca45b

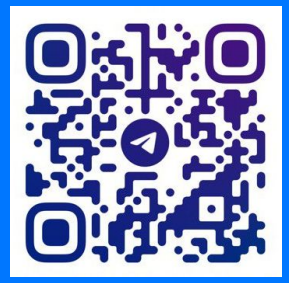
Найти все анаграммы в строке (medium)

- Дан текст `text` и образец `pattern`;
- Найти все анаграммы образца `pattern` в тексте `text`

```
std::vector<int> findAnagrams(const std::string &text, const string &pattern);
```

Ссылка:

https://interview.cups.online/live-coding/?room=d0bde9ed-70ad-4d50-a3a8-c13a9b4d3a3e



Спасибо
за внимание!