

Углублённое программирование на C++

Ретроспектива пройденных тем

Кухтичев Антон



education

10 июля 2024 года

Содержание занятия

- Пароли (**easy**)
- Общий префикс (**easy**)
- Валидные скобки I (**easy**)
- Валидные скобки II (**easy**)
- Слияние двух односвязных списков (**easy**)
- Валидная анаграмма (**easy**)
- Найти все анаграммы в строке (**medium**)
- Проверить, что бинарное дерево это бинарное дерево поиска (**medium**)
- Самая длинная подстрока с уникальными символами (**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>

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>

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=e1c2b34b-4982-4b86-8d35-5aa3e7967fdf>

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=b9a17468-6174-4653-aa97-e9a351bc90e2>

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>

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=ac0eb710-82e0-4ebe-ba2f-1009252a4c79>

Самая длинная подстрока с уникальными символами (medium)

- Дана строка str
- Найти самую длинную подстроку с уникальными символами

```
int lengthOfLongestSubstring(const std::string& str)
```

Ссылка:

<https://interview.cups.online/live-coding/?room=4a5ad2e6-ae9b-4311-aca9-bc1cea01c50d>

Сумма всех чисел от корня-до-листа (medium)

- Дан указатель на корень;
- Найти сумму, полученную сложением всех чисел, получаемых от корня до листа

```
int sumNumbers(TreeNode* root);
```

Ссылка:

<https://interview.cups.online/live-coding/?room=389ccf33-6a3a-451d-b391-3bc52d51e1b3>



Спасибо
за внимание!