Модуль 1 Перебор
Лекция 1.4 Правильные скобочные последовательности

На этой лекции мы будем перебирать *правильные скобочные последовательности*. Это последовательности из одинакового числа открывающихся и закрывающихся скобок, которые могут встретиться в корректном арифметическом выражении. Ниже приведены несколько примеров правильных скобочных последовательностей, которые получаются из арифметических выражений удалением чисел и плюсов.

$$\begin{aligned}(1+1)+(1+1) &\quad\to\quad ()()\\ ((1+1)+1)+(1+1) &\quad\to\quad (())()\\ ((1+1)+(1+1)) &\quad\to\quad (()())\end{aligned}$$

Далее идут примеры неправильных скобочных последовательностей.

$$\begin{aligned})(\\ (()((\\ ())(()\end{aligned}$$

Данное определение неудобно использовать для проверки скобочной последовательности на правильность. Нужен легко проверяемый критерий. Заметим, что на любом начальном отрезке (или, как говорят, на *префиксе*) правильной скобочной последовательности число открывающихся скобок не меньше числа закрывающихся. Для неправильных последовательностей это условие нарушается. Для последовательности «)(» оно нариушается уже на префиксе длины 1 — последовательность начинается с закрывающейся скобки. В последовательности «())(()» среди первых трёх скобок больше закрывающихся, чем открывающихся. У последовательности «(()((» другая проблема — в ней разные количества открывающихся и закрывающихся скобок.

Разберём функцию на рис. 1, которая проверяет скобочную последовательность на правильность.

Функции передаётся строка s из открывающихся и закрывающихся скобок.

bool correct(string s)

Переменная bal — это *баланс* — разность между числом открывающихся и закрывающихся скобок. Вначале баланс равен нулю.

int bal = 0;

Будем идти по строке циклом и пересчитывать баланс.

for (**int** i = 0; i < (**int**)s.size(); i++)

Если встретилась открывающаяся скобка, баланс увеличивается на 1.

```
bool correct(string s)
{
    int bal = 0;
    for (int i = 0; i < (int)s.size(); i++)
    {
        if (s[i] == '(')
            bal++;
        else
            bal--;
        if (bal < 0)
            return false;
    }
    return (bal == 0);
}
```

Рис. 1. Проверка скобочной последовательности на правильность

if (s[i] == ‘(’)
    bal++;

Если закрывающаяся — уменьшается на 1.

else
    bal−−;

Если баланс отрицательный — критерий нарушился, и наша скобочная последовательность неправильная. Функция возвращает значение false.

if (bal < 0)
    return false;

В конце мы проверяем, что баланс равен нулю, то есть общее число открывающихся скобок совпадает с числом закрывающихся.

return (bal == 0);

Упражнение 1.4.1
Сколько существует правильных скобочных последовательностей из трёх открывающихся и трёх закрывающихся скобок?

Функция на рис. 2 перебирает все правильные скобочные последовательности из $2n$ скобок.

В символьном векторе s будет строиться скобочная последовательность.

vector<**char**> s;

В функцию rec передаются два аргумента: idx — индекс позиции в векторе s, в которую мы будем записывать очередную скобку; bal — текущий баланс.

void rec(**int** idx, **int** bal)

Изначально функция rec вызывается от нулевых значений обоих аргументов.

rec(0, 0);

```
void rec(int idx, int bal)
{
    if (idx == 2 * n)
    {
        if (bal == 0)
            out();
        return;
    }

    s[idx] = '(';
    rec(idx + 1, bal + 1);
    if (bal == 0)
        return;
    s[idx] = ')';
    rec(idx + 1, bal - 1);
}
```

Рис. 2. Перебор правильных скобочных последовательностей

Вместо цикла в функции rec идёт перебор двух вариантов. Сначала мы ставим в текущую позицию s[idx] открывающуюся скобку:

s[idx] = ‘(’;

И делаем рекурсивный переход к следующей позиции.

rec(idx + 1, bal + 1);

При этом баланс увеличивается на 1.

Закрывающуюся скобку мы можем поставить, только если баланс положительный. Если он равен нулю, то нам не надо рассматривать второй вариант, и мы выходим из функции.

if (bal == 0)
    return;

Отрицательного баланса мы не допускаем. Итак, если баланс положительный, мы ставим закрывающуюся скобку и делаем рекурсивный переход, уменьшая баланс на 1.

s[idx] = ’)’;
rec(idx + 1, bal - 1);

Рассмотрим условие выхода из рекурсии. Если мы поставили $2n$ скобок, то мы проверяем, нулевой ли у нас баланс.

if (idx == 2 * n)
    if (bal == 0)

Если да, то выводим содержимое вектора:

out();

Данная функция out() будет отличаться от той функции, которая была в предыдущих программах.

В любом случае выполняем выход из функции rec:

return;

Функция rec переберёт все правильные скобочные последовательности в лексикографическом порядке, потому что она всегда сначала пытается ставить открывающуюся скобку, а потом — закрывающуюся.