1. String to Integer (atoi)

第一次做，考虑到有str中会存在字母，符号等字符所以采用了正则表达式。并且使用了Double.parseDoulbe(str).结果运行时间较长，不理想。

|  |
| --- |
| import java.util.regex.Matcher;  import java.util.regex.Pattern;  public class Solution {  public int myAtoi(String str) {  int result = 0;  String dest = "";  Matcher a = Pattern.compile("^\\s\*?((\\d+)|(-\\d+)|(\\+\\d+))").matcher(str);  if(a.find()){  System.out.println(a.group(1));  result = tools(a.group(1));  }  return result;  }  private int tools(String str){  long result = 0;  try{  result=Long.parseLong(str);  }catch(NumberFormatException e){  e.printStackTrace();  }  if(result > Integer.MAX\_VALUE){  return Integer.MAX\_VALUE;  }  if(result < Integer.MIN\_VALUE){  return Integer.MIN\_VALUE;  }  return (int)result;  }  public static void main(String[] args){  String str = "2147483648";  Solution s = new Solution();  System.out.println(s.myAtoi(str));    }  } |

网上代码，简单。没有考虑更为复杂的情况，时间较短。

|  |
| --- |
| **if** (str == **null** || str.length() < 1)  **return** 0;    *// trim white spaces*  str = str.trim();    **char** flag = '+';    *// check negative or positive*  **int** i = 0;  **if** (str.charAt(0) == '-') {  flag = '-';  i++;  } **else** **if** (str.charAt(0) == '+') {  i++;  }  *// use double to store result*  **double** result = 0;    *// calculate value*  **while** (str.length() > i && str.charAt(i) >= '0' && str.charAt(i) <= '9') {  result = result \* 10 + (str.charAt(i) - '0');  i++;  }    **if** (flag == '-')  result = -result;    *// handle max and min*  **if** (result > Integer.MAX\_VALUE)  **return** Integer.MAX\_VALUE;    **if** (result < Integer.MIN\_VALUE)  **return** Integer.MIN\_VALUE;    **return** (**int**) result;} |