实训4：日志文件解析与分析

· ***目标***：掌握正则表达式和数据聚合。

· ***任务***：解析服务器日志，统计IP和URL访问频率。

· ***详细要求***：

o 读取日志文件（server.log），格式如192.168.1.1 - [2025-05-01 10:00:00] "GET /page1.html ...".

o 使用正则表达式提取IP、时间、URL。

o 统计每个IP的访问次数和每个URL的访问频率。

o 输出前10高频IP和URL到log_stats.txt，格式为“IP/URL: 次数”。

o 验证日志格式，跳过无效行并记录日志。

o 添加时间过滤功能（如仅分析某天的数据）。

o 保存分析结果，支持追加写入。

技能：正则表达式、Counter、文件I/O、日志记录。

```
# -*- coding: utf-8 -*-
import re
from collections import Counter
from datetime import datetime
import logging

# 设置日志记录
logging.basicConfig(filename='log_parser_debug.log', level=logging.INFO, format='%(asctime)s - %(message)s')

# 正则模式：提取 IP、时间、URL
LOG_PATTERN = re.compile(r'(?P<ip>\d+\.\d+\.\d+\.\d+) - \[(?P<time>[\d\-:\s]+)\] "\w+ (?P<url>/[^\s]*)')

def parse_log(file_path, filter_date=None):
    ip_counter = Counter()
    url_counter = Counter()
    total_lines = 0
    valid_lines = 0

    with open(file_path, 'r', encoding='utf-8') as f:
        for line in f:
            total_lines += 1
            match = LOG_PATTERN.search(line)
            if not match:
                logging.warning(f"无效日志行: {line.strip()}")
                continue

            ip = match.group('ip')
            time_str = match.group('time')
            url = match.group('url')

            # 时间过滤
            if filter_date:
```

```
                log_date = datetime.strptime(time_str, '%Y-%m-%d %H:%M:%S').date()
                if str(log_date) != filter_date:
                    continue

            ip_counter[ip] += 1
            url_counter[url] += 1
            valid_lines += 1

    logging.info(f"处理完成，共 {total_lines} 行，有效日志行 {valid_lines} 行")
    return ip_counter.most_common(10), url_counter.most_common(10)

def save_stats(ip_stats, url_stats, output_file):
    with open(output_file, 'a', encoding='utf-8') as f:
        f.write('Top 10 IPs:\n')
        for ip, count in ip_stats:
            f.write(f'{ip}: {count}\n')
        f.write('\nTop 10 URLs:\n')
        for url, count in url_stats:
            f.write(f'{url}: {count}\n')
        f.write('\n')

if __name__ == "__main__":
    log_file = 'server.log'
    output_file = 'log_stats.txt'
    date_filter = input("请输入要分析的日期 (格式 YYYY-MM-DD)，留空表示不过滤: ").strip()
    if date_filter == '':
        date_filter = None

    ip_stats, url_stats = parse_log(log_file, date_filter)
    save_stats(ip_stats, url_stats, output_file)
    print("分析完成，结果已保存至 log_stats.txt")

```