图像压缩问题 - 动态规划算法实现

作业题目B：图像压缩问题
学号尾数：8

📋 项目简介

本项目使用动态规划算法求解图像压缩问题，通过最优分段方案使灰度序列的总存储位数最小。项目包含完整的算法实现和精美的可视化动画。

问题描述

给定长度为 n 的灰度值序列，每个值范围 [0, 255]，需要将序列分段存储。每段的存储公式为：

```
存储位数 = 8 + l + b*l
```

其中：

- `l`：段长度（≤ 255）
- `b`：该段所需编码位数 = $\lceil \log_2(\max-\min+1) \rceil$

目标：找到最优分段方案，使总存储位数最小。

🚀 快速开始

环境要求

- Python 3.7+
- 操作系统：Windows / macOS / Linux

依赖安装

```
pip install numpy matplotlib
```

或使用 requirements.txt：

```
pip install -r requirements.txt
```

一键运行

```
python image_compression.py
```

程序将自动：

1. 生成测试序列（长度150，随机种子42）
2. 执行动态规划算法
3. 生成可视化动画（image_compression.gif）
4. 输出结果到控制台和文件

预计运行时间：4-8分钟

📊 输出文件

运行完成后将生成以下文件：

1. **image_compression.gif** - 可视化动画（约20-30MB）
   - 分辨率：1600x1000
   - 帧率：15 FPS
   - 时长：约30-50秒
2. **sequence_data.txt** - 序列数据和结果
   - 包含随机种子、序列数据
   - 最优分段方案详情
   - 压缩统计信息

📈 测试数据

数据生成

使用 `generate_test_sequence()` 函数生成：

- **序列长度**：150（满足 n ≥ 120）
- **随机种子**：42（可复现）

数据特点

✅ 包含多个波动区段
✅ 避免单调序列
✅ 灰度分布适中
✅ 每段长度 ≤ 255

自定义数据

修改 `main()` 函数中的参数：

```
SEQUENCE_LENGTH = 150     # 修改序列长度
RANDOM_SEED = 42          # 修改随机种子
```

📝 文件结构

```
project/
│
├── image_compression.py      # 主程序（算法+可视化）
├── README.md                 # 本文档
├── requirements.txt          # 依赖列表
├── report.md                 # 报告
├── image_compression.gif     # 输出动画（运行后生成）
└── sequence_data.txt         # 输出数据（运行后生成）
```