Assignment #C: bfs & dp

Updated 1436 GMT+8 Nov 25, 2025

2025 fall, Complied by 苗越 数学科学学院

说明：

1）请把每个题目解题思路（可选），源码 Python, 或者 C++（已经在 Codeforces/Openjudge 上 AC），截图（包含 Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用 word）。AC 或者没有 AC，都请标上每个题目大致花费时间。

2）提交时候先提交 pdf 文件，再把 md 或者 doc 文件上传到右侧"作业评论"。Canvas 需要有同学清晰头像、提交文件有 pdf、"作业评论"区有上传的 md 或者 doc 附件。

3）如果不能在截止前提交作业，请写明原因。

1. 题目

sy321 迷宫最短路径

bfs, https://sunnywhy.com/sfbj/8/2/321

思路：

代码：

```
n, m = map(int, input().split())
matrix = []
for _ in range(n):
    matrix.append(list(map(int, input().split())))
direction = [(-1, 0), (1, 0), (0, -1), (0, 1)]
visited = [[False for _ in range(m)] for _ in range(n)]
parent_lis = [[None for _ in range(m)] for _ in range(n)]
queue = [(0, 0)]
visited[0][0] = True
while queue:
    i = queue.pop(0)
    if matrix[i[0]][i[1]] == 1:
        continue
    else:
        for dx, dy in direction:
            nx, ny = i[0] + dx, i[1] + dy
            if 0 <= nx < n and 0 <= ny < m and not visited[nx][ny] and matrix[nx]
[ny] == 0:
                queue.append((nx, ny))
                visited[nx][ny] = True
                parent_lis[nx][ny] = i
                if nx == n-1 and ny == m-1:
                    break
```

```
            if nx == n-1 and ny == m-1:
                break
    path_lis = [(n-1, m-1)]
    current_node = (n-1, m-1)
    while True:
        if parent_lis[current_node[0]][current_node[1]] is None:
            break
        path_lis.append(parent_lis[current_node[0]][current_node[1]])
        current_node = parent_lis[current_node[0]][current_node[1]]
    path_lis.reverse()
    for i in path_lis:
        print(i[0]+1, i[1]+1)
```

代码运行截图（至少包含有"Accepted"）

sy324 多终点迷宫问题

bfs, https://sunnywhy.com/sfbj/8/2/324

思路：

代码：

```
from collections import deque
def bfs_maze(maze, n, m):
    # Initialize the result grid with -1
    result = [[-1 for _ in range(m)] for _ in range(n)]

    # Directions for moving up, down, left, right
    directions = [(-1, 0), (1, 0), (0, -1), (0, 1)]

    # BFS initialization
    queue = deque()

    # Start from the top-left corner if it's a path
    if maze[0][0] == 0:
        queue.append((0, 0))
        result[0][0] = 0

    while queue:
        x, y = queue.popleft()

        for dx, dy in directions:
            nx, ny = x + dx, y + dy

            # Check if the new position is within bounds and is a path
            if 0 <= nx < n and 0 <= ny < m and maze[nx][ny] == 0:
                # If the cell has not been visited yet
                if result[nx][ny] == -1:
                    result[nx][ny] = result[x][y] + 1
                    queue.append((nx, ny))

    return result

n, m = map(int, input().split())
maze = [list(map(int, input().split())) for _ in range(n)]
result = bfs_maze(maze, n, m)
for row in result:
    print(' '.join(map(str, row)))
```

代码运行截图（至少包含有"Accepted"）

M02945: 拦截导弹

dp, greedy http://cs101.openjudge.cn/pctbook/M02945

思路：

代码：

```
n = int(input())
missiles = list(map(int, input().split()))
if n == 0:
    print(0)
else:
    dp_table = [1] * n
    for i in range(1, n):
        for j in range(i):
            if missiles[i] <= missiles[j]:
```

```
                dp_table[i] = max(dp_table[i], dp_table[j] + 1)
    print(max(dp_table))
```

代码运行截图 （至少包含有"Accepted"）

189A. Cut Ribbon

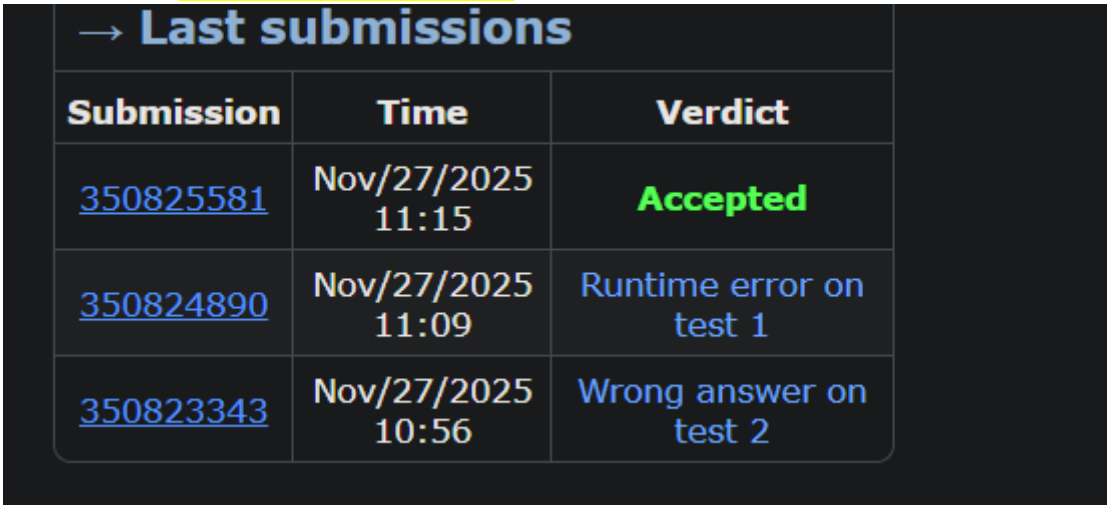
brute force/dp, 1300, https://codeforces.com/problemset/problem/189/A

思路：

代码：

```
n, a, b, c = map(int, input().split())

# Initialize DP table: dp[i] = maximum number of pieces for length i
# Use -1 to indicate "not possible"
dp = [-1] * (n + 1)
dp[0] = 0  # Base case: 0 pieces for length 0

for i in range(1, n + 1):
    # Try cutting a piece of length a
    if i >= a and dp[i - a] != -1:
        dp[i] = max(dp[i], dp[i - a] + 1)

    # Try cutting a piece of length b
    if i >= b and dp[i - b] != -1:
        dp[i] = max(dp[i], dp[i - b] + 1)

    # Try cutting a piece of length c
    if i >= c and dp[i - c] != -1:
        dp[i] = max(dp[i], dp[i - c] + 1)

print(dp[n])
```

代码运行截图（至少包含有"Accepted"）



→ Last submissions

Submission	Time	Verdict
350825581	Nov/27/2025 11:15	Accepted
350824890	Nov/27/2025 11:09	Runtime error on test 1
350823343	Nov/27/2025 10:56	Wrong answer on test 2

M01384: Piggy-Bank

dp, http://cs101.openjudge.cn/practice/01384/

思路：

代码：

```
cases = int(input())
for _ in range(cases):
    piggy, total = map(int, input().split())
    total -= piggy
    types = int(input())
    value_lis = []
    weight_lis = []
    for _ in range(types):
        value, weight = map(int, input().split())
        value_lis.append(value)
        weight_lis.append(weight)
    dp_table = [float('inf')] * (total + 1)
    dp_table[0] = 0
    for i in range(1, total + 1):
        for j in range(types):
            if i >= weight_lis[j] and dp_table[i - weight_lis[j]] != float('inf'):
                dp_table[i] = min(dp_table[i], dp_table[i - weight_lis[j]] +
value_lis[j])
    if dp_table[total] == float('inf'):
        print("This is impossible.")
    else:
        print(f"The minimum amount of money in the piggy-bank is
{dp_table[total]}.")
```

代码运行截图（至少包含有"Accepted"）

M02766: 最大子矩阵

dp, kadane, http://cs101.openjudge.cn/pctbook/M02766

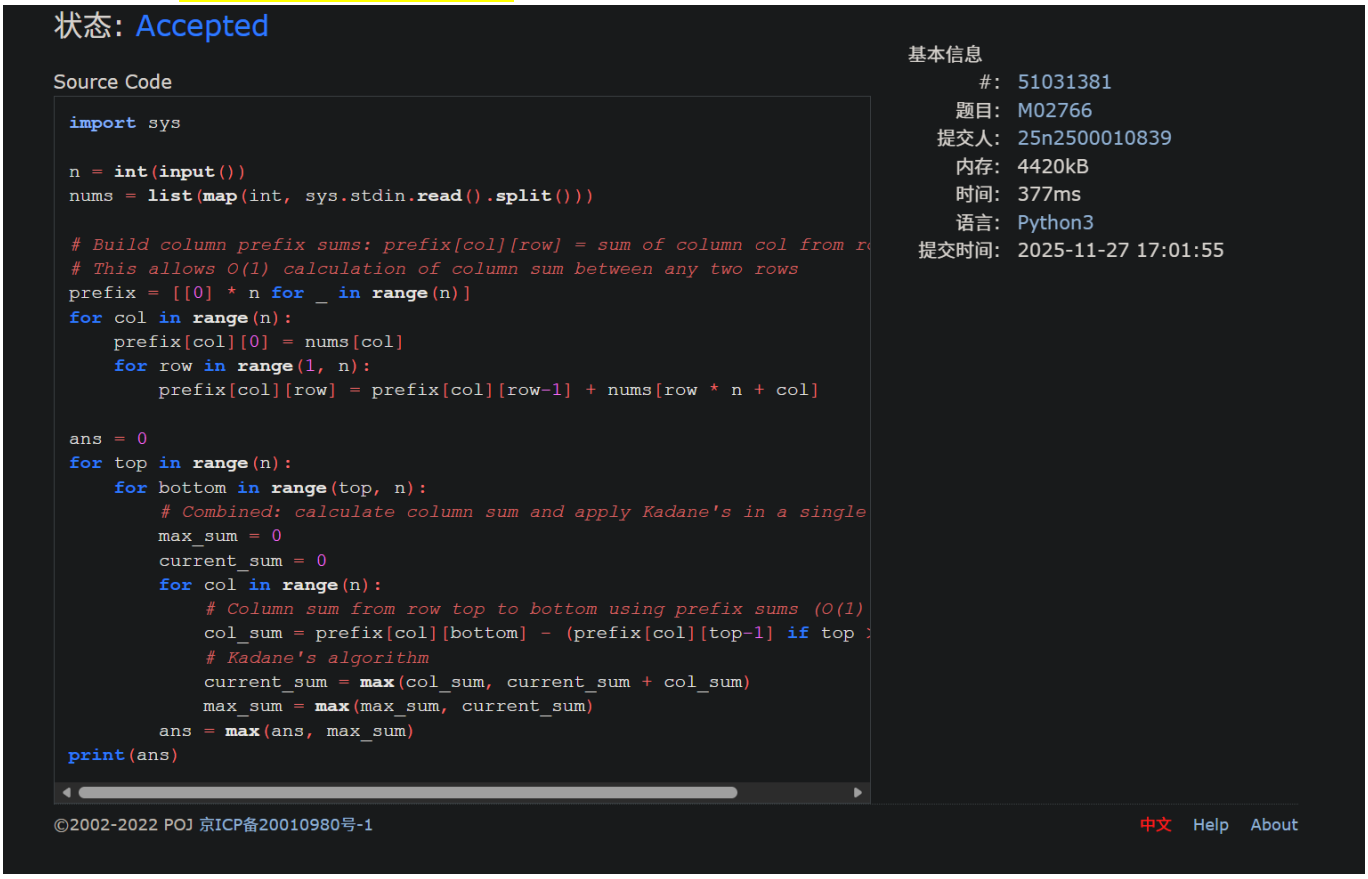
思路：

代码：

```
import sys

n = int(input())
nums = list(map(int, sys.stdin.read().split()))

# Build column prefix sums: prefix[col][row] = sum of column col from row 0 to row
# This allows O(1) calculation of column sum between any two rows
prefix = [[0] * n for _ in range(n)]
for col in range(n):
    prefix[col][0] = nums[col]
    for row in range(1, n):
        prefix[col][row] = prefix[col][row-1] + nums[row * n + col]

ans = 0
for top in range(n):
    for bottom in range(top, n):
        # Combined: calculate column sum and apply Kadane's in a single pass
        max_sum = 0
        current_sum = 0
        for col in range(n):
            # Column sum from row top to bottom using prefix sums (O(1) operation)
            col_sum = prefix[col][bottom] - (prefix[col][top-1] if top > 0 else 0)
            # Kadane's algorithm
```

```
            current_sum = max(col_sum, current_sum + col_sum)
            max_sum = max(max_sum, current_sum)
        ans = max(ans, max_sum)
print(ans)
```

代码运行截图 （至少包含有"Accepted"）



2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“计概 2024fall 每日选做”、CF、LeetCode、洛谷等网站题目。

choose_the_player:

the transfer equation is fairly reasonable, but it's hard to generate. Just remember in the future if you are given multiple choice in one dp problem regarding i, then maybe your dp table should contain more than one number for each i.

boredom:

i learned Counter, which is capable of quickly counting the appearing time of certain element. It's all about downsize the huge sequence-noticing what we need to solve has nothing to do with order. maze_easiest:

we can store step and visit state in one matrix. this is super easy, making it no need to track steps in queue. by the way, deque is a good tool when solving those queue or stack questions.

number_operation:

in this problem, we can also store steps and visit state in one long list. the most important thing is to realize it's a bfs question. to think reversely, storing steps in queue is also reasonable in maze.

vacation:

setting the first case for dp table is also important. must do it right lol. dp table too.

wealthy_ppl:

i cannot find the solution by myself. it adopts a similar method to kadane but... the discard one strategy is tricky. somehow i feel it's like differenciation equation in math, it's harder to generate it than guessing it.

pots:

i suddenly wonder how openjudge identify presentation error. meanwhile, this program can still be optimized, if i redesign visited, however i don't know how to achieve this. i mean, store integer in visited.