

# assignmentC

## Assignment #C : bfs & dp

Updated 1436 GMT+8 Nov 25, 2025

2025 fall, Compiled by 顾桂榕 基础医学院



## 顾桂榕 医学预科办

### 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:

```

from collections import deque
n,m = map(int,input().split())
maze = []
visited = [[False] * m for _ in range(n)]
for _ in range(n):
    maze.append(list(map(int,input().split())))
def escape():
    directions = [(0,1),(0,-1),(1,0),(-1,0)]
    queue = deque()
    queue.append((0,0,[(1,1)]))
    visited[0][0] = True
    while queue:
        x,y,path = queue.popleft()

        if x==n-1 and y==m-1:
            return path
        for dx,dy in directions:
            nx,ny = x+dx,y+dy
            if 0<=nx<n and 0<=ny<m:
                if maze[nx][ny]!=1:
                    if not visited[nx][ny]:
                        visited[nx][ny]=True
                        new_path = path+[(nx+1,ny+1)]
                        queue.append((nx,ny,new_path))

res = escape()
for _ in res:
    print(f'_{[0]}'+' '+f'_{[1]}')

```

代码运行截图 (至少包含有"Accepted")

打开/关闭工作区窗口

『浙大、复旦、上交、华师、中科大计算机&软件』等上机

RoseRong

难度

代码书写

```
2 n,m = map(int,input().split())
3 maze = []
4 visited = [[False] * m for _ in range(n)]
5 for _ in range(n):
6     maze.append(list(map(int,input().split())))
7 def escape():
8     directions = [(0,1),(0,-1),(1,0),(-1,0)]
9     queue = deque()
10    queue.append((0,0,[(1,1)]))
11    visited[0][0] = True
12    while queue:
13        x,y,path = queue.popleft()
14
15        if x==n-1 and y==m-1:
16            return path
17        for dx,dy in directions:
18            nx,ny = x+dx,y+dy
```

🏠 个人主页

☆ 我的收藏

📊 题目统计

🎨 深色主题

💡 意见反馈

🚪 退出登录

测试输入

提交结果

历史提交

完美通过

[查看题解](#)

100% 数据通过测试 [详情](#)

运行时长: 0 ms

sy324多终点迷宫问题

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

思路:

代码:

```
from collections import deque
n,m = map(int,input().split())
maze = []
for _ in range(n):
    maze.append(list(map(int,input().split())))
directions = [(0,1),(0,-1),(1,0),(-1,0)]
def escape(e_x,e_y):
    visited = [[False] * m for _ in range(n)]
    queue = deque()
    queue.append((0, 0, 0))
    visited[0][0] = True
    while queue:
        x,y,step = queue.popleft()
        if x==e_x and y==e_y:
            return step
        for dx,dy in directions:
            nx,ny = x+dx,y+dy
            if (0<=nx<n and 0<=ny<m) and maze[nx][ny]!=1 and (not
visited[nx][ny]):
                visited[nx][ny] = True
                queue.append((nx,ny,step+1))
    return -1
res = []
for i in range(n):
    res.append([])
    for j in range(m):
        res[-1].append(escape(i,j))
for k in range(n):
    print(*res[k])
```

代码运行截图 (至少包含有"Accepted")

## 6 考研算法全程训练营

『浙大、复旦、上交、华师、中科大计算机&软件』等上机

代码书写

```
2 n,m = map(int,input().split())
3 maze = []
4 for _ in range(n):
5     maze.append(list(map(int,
6 directions = [(0,1),(0,-1),(1,
7 def escape(e_x,e_y):
8     visited = [[False] * m for
9     queue = deque()
10    queue.append((0, 0, 0))
11    visited[0][0] = True
12    while queue:
13        x,y,step = queue.popleft()
14        if x==e_x and y==e_y:
15            return step
16        for dx,dy in directions:
17            nx,ny = x+dx,y+dy
18            if (0<=nx<n and 0<=ny<m) and maze[nx
```

测试输入 提交结果 历史提交

RoseRong

难度

个人主页

我的收藏

题目统计

深色主题

意见反馈

退出登录

完美通过

[查看题解](#)

100% 数据通过测试 [详情](#)

运行时长: 729 ms

**M02945: 拦截导弹**

*dp, greedy* <http://cs101.openjudge.cn/pctbook/Mo2945>

思路:

想了好久

改了两版*code*

今天不想看了, 明天再来

代码:

```
def max_intercepted_missiles(k, heights):
    # Initialize the dp array
    dp = [1] * k

    # Fill the dp array
    for i in range(1, k):
        for j in range(i):
            if heights[i] <= heights[j]:
                dp[i] = max(dp[i], dp[j] + 1)

    # The result is the maximum value in dp array
    return max(dp)

if __name__ == "__main__":

    k = int(input())
    heights = list(map(int, input().split()))

    result = max_intercepted_missiles(k, heights)
    print(result)
```

## 代码运行截图 (至少包含有"Accepted")

#51021278提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```
def max_intercepted_missiles(k, heights):
    # Initialize the dp array
    dp = [1] * k

    # Fill the dp array
    for i in range(1, k):
        for j in range(i):
            if heights[i] <= heights[j]:
                dp[i] = max(dp[i], dp[j] + 1)

    # The result is the maximum value in dp array
    return max(dp)

if __name__ == "__main__":
    k = int(input())
    heights = list(map(int, input().split()))

    result = max_intercepted_missiles(k, heights)
    print(result)
```

基本信息

#: 51021278  
题目: M02945  
提交人: R.  
内存: 5896kB  
时间: 28ms  
语言: Python3  
提交时间: 2025-11-26 21:35:25

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

## 189A. Cut Ribbon

brute force/dp, 1300,

<https://codeforces.com/problemset/problem/189/A>

思路:

一开始的思路就是照着小偷背包的来想, 代码没怎么变, 就是把每一段的长度看成 *weight*, 每一段的价值看成1

但是WA了, 很疑惑

意识到 $a, b, c$ 可以重复使用, 因此我们的状态转移方程的后一项应该是当行的剩余容量能装的*value*

还是不对

试验了7 5 5 2

发现这一代码并没有考虑*ribbon*都取尽, *print3*说明全用2, 但是会剩下1

看题解发现不用二维数组, 用1维即可

初始化所有值为一个很大的负数

如果 $\geq a$ 且为 $a$ 的倍数, 那么在原来基础上+1

若不为 $a$ 的倍数, 那么相当于一个很负的数稍微不那么负

每次遍历 $a, b, c$ 三个数

还是这一组数据

那么到5的时候，就是1，即长度为5时刚好切完的数量

代码：

```
n,a,b,c = map(int,input().split())
dp = [-10000]*(n+1)
dp[0]=0

for i in range(1,n+1):
    if i>=a:
        dp[i]=max(dp[i],dp[i-a]+1)
    if i>=b:
        dp[i]=max(dp[i],dp[i-b]+1)
    if i>=c:
        dp[i]=max(dp[i],dp[i-c]+1)
print(dp[n])
```

代码运行截图（至少包含有"Accepted"）

By Rose-Rong, contest: Codeforces Round 119 (Div. 2), problem: (A) Cut Ribbon, **Accepted**, #, [Copy](#)

```
n, a, b, c = map(int, input().split())
dp = [-10000] * (n + 1)
dp[0] = 0

for i in range(1, n + 1):
    if i >= a:
        dp[i] = max(dp[i], dp[i - a] + 1)
    if i >= b:
        dp[i] = max(dp[i], dp[i - b] + 1)
    if i >= c:
        dp[i] = max(dp[i], dp[i - c] + 1)
print(dp[n])
```

## Mo1384: Piggy-Bank

dp, <http://cs101.openjudge.cn/practice/o1384/>

思路：

这题明显和上一题很像

学完上一题直接copy发现TLE

原来有更快的方法

我们用硬币来问，每一次更新当前硬币种类的最优解

一是能从硬币质量往后，省去了前面的讨论

二是当 $dp[j-w] == 10^{**}9$ 时说明这个值没有更新过，所以肯定不是我们要找的答案，

跳过

遂AC

代码：

```
T = int(input())
for _ in range(T):
    E,F = map(int,input().split())
    if E>F:
        print('This is impossible.')
    else:
        total = F-E
        N = int(input())
        coins = []
        for i in range(N):
            v,w = map(int,input().split())
            coins.append((v,w))

        dp = [10**9]*(total+1)
        dp[0]=0
        for v,w in coins:
            for j in range(w,total+1):
                if dp[j-w]!=10**9:
                    dp[j]=min(dp[j],dp[j-w]+v)
        if dp[total]==10**9:
            print('This is impossible.')
        else:
            print(f'The minimum amount of money in the piggy-bank is
{dp[total]}.'
```

## 代码运行截图 (至少包含有"Accepted")

#51016631提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```
T = int(input())
for _ in range(T):
    E, F = map(int, input().split())
    if E > F:
        print('This is impossible.')
    else:
        total = F - E
        N = int(input())
        coins = []
        for i in range(N):
            v, w = map(int, input().split())
            coins.append((v, w))

        dp = [10**9] * (total + 1)
        dp[0] = 0
        for v, w in coins:
            for j in range(w, total + 1):
                if dp[j - w] != 10**9:
                    dp[j] = min(dp[j], dp[j - w] + v)
        if dp[total] == 10**9:
            print('This is impossible.')
        else:
            print(f'The minimum amount of money in the piggy-bank is {dp[total]}')
```

基本信息

#: 51016631  
题目: 01384  
提交人: R.  
内存: 3996kB  
时间: 8544ms  
语言: Python3  
提交时间: 2025-11-26 17:50:40

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

## Mo2766: 最大子矩阵

*dp*, *kadane*, <http://cs101.openjudge.cn/pctbook/Mo2766>

思路:

以前做过了

*kadane*算法还是很巧妙的

二维先压缩成一维再用*kadane*算法

代码:

```
def submatrix_sum(matrix, N):
    def kadane(arr):
        max_end_here = max_so_far = arr[0]
        for x in arr[1:]:
            max_end_here = max(x, x + max_end_here)
            max_so_far = max(max_so_far, max_end_here)
        return max_so_far

    max_sum = float('-inf')

    for top in range(N):
```

```

        compressed_matrix = [0] * N
        for bottom in range(top,N):
            for col in range(N):
                compressed_matrix[col] += matrix[bottom][col]
            max_sum = max(kadane(compressed_matrix),max_sum)
    return max_sum

```

```

import sys
data = sys.stdin.read().split()
N = int(data[0])
numbers = list(map(int,data[1:]))
matrix = [numbers[i*N:(i+1)*N] for i in range(N)]
ans = submatrix_sum(matrix,N)
print(ans)

```

代码运行截图 (至少包含有"Accepted")

#51018775提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: Accepted

源代码

```

def max_submatrix(matrix, n):
    def kadane(arr):
        max_end_here = max_so_far = arr[0]
        for x in arr[1:]:
            max_end_here = max(x, max_end_here + x)
            max_so_far = max(max_so_far, max_end_here)
        return max_so_far

    max_sum = float('-inf')

    for top in range(n):
        temp_col_num = [0] * n
        for bottom in range(top, n):
            for col in range(n):
                temp_col_num[col] += matrix[bottom][col]
            max_sum = max(max_sum, kadane(temp_col_num))
    return max_sum

import sys
data = sys.stdin.read().split()
n = int(data[0])
numbers = list(map(int, data[1:]))
matrix = [numbers[i * n:(i + 1) * n] for i in range(n)]

max_sum = max_submatrix(matrix, n)
print(max_sum)

```

基本信息

#: 51018775  
 题目: M02766  
 提交人: R.  
 内存: 4564kB  
 时间: 209ms  
 语言: Python3  
 提交时间: 2025-11-26 19:54:59

©2002-2022 POJ 京ICP备20010980号-1

[English](#) [帮助](#) [关于](#)

## 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。

之前模考学过了 *bfs*. 有些淡忘但拾起来还是很快的  
*dp* 感觉还是不太会，把经典题目题解记住再多做些题吧  
事情又多起来了，加油！