

**程序设计方法与艺术**

解

题

报

告

班 级：物联网工程22-2班

组 长：20222419 XXX

组 员：20222421 XXX

20222424 XXX

20222427 XXX

**题目A** **植被保护**

题面:

植被覆盖是地球上生命存在和繁荣的基础之一。它在生态系统中扮演着 重要的角色，对环境的健康和可持续性具有深远影响。植被覆盖对维持生物 多样性至关重要。植物为各种动物提供栖息地和食物源。植物的多样性直接 关系到其他生物的多样性。保护植被覆盖有助于维持生态平衡，保护珍稀和 濒危物种。植被覆盖还对空气和水质的净化起着重要作用。植物通过吸收空气中的有害气体和颗粒物，改善空气质量。树木的根系能够过滤和吸收地下水中的污染物，提高水质。这对于维护健康的生态系统、保护人类健康至关重要。H市现在正在大力发展工业，众所周知发展工业，会影响当地的植被覆盖率。

第一年 H 市的植被覆盖数为 N 平方千米，当植被覆盖数达到 M 平方千米以下时，则说明当地已经严重污染，植被覆盖不足。假设 H 市的植被覆盖数每年以K %的减少。请问多少年之后 H 市将会严重污染？

输入说明：一行包含三个整数 N,M,K ，N 表示第一年 H 市的植被覆盖数,M 表示 M 平方千米以下时，则说明当地已经严重污染，K 表示 H 市的植被覆盖数每年以 K%的减少

输出说明：输出一个整数，表示多少年之后 H 市将会严重污染。

解题思路：

这是高中常见的次方问题，难点是如何用代码实现

首先将每年以 K%的减少转化为倍率：t=(100 - K)/100，

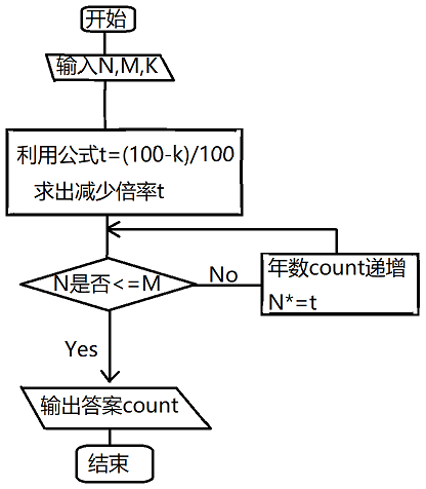
然后再使用死循环计算今年数量N减少到小于目标数量M所乘以 倍率t的次数即可.

注意事项：

数据N,M,K都应以浮点型存储（整型会丢失小数部分）

到达目标值应以break结束循环

基于上述解题思路给出的流程图为：



**运行结果：**







**题目C** 联合密码

题面:

环保部门在城市工业区设置了环境污染物检测和预警实验室，实验室会收集 工业区附近的地表及地下水系统的样本并进行检测，同时保存过往的数据用于比 对。因此实验室安装了一个特殊的门禁系统，每隔一段时间就会进行调整，避免 工作人员不小心对外透露相关信息。 这款门禁系统的密码是若干个公式，每次使用 2 个，然后按照要求求出指定 的结果，小李今天拿到了其中 2 个，其中一个是 (x^2+a)^0.5，另一个是((b‐x) ^2+1)^0.5。a 和 b 的信息会显示在屏幕上，今天的要求是求两个公式和的最小值。 输入说明： 输入 a、b,两个非负实数 输出说明： 输出表达式的最小值，精确到小数点后 6 位。

输入样例： 4.000    4.000

输出样例： 5.000000

解决思路:

这个问题可以通过求导来解决。首先，我们可以将两个公式相加，得到一个新的公式：

f(x) = (x2+a)0.5 + ((b-x)2+1)0.5。然后，我们对这个新的公式求导，得到f’(x) = x/((x2+a)0.5) - (b-x)/(((b-x)2+1)0.5)。接下来，我们可以找到f’(x) = 0的解，即x = (a(b2+1))/(a+b2+1)。最后，我们将这个解代入原来的公式中，就可以得到两个公式和的最小值。

例如，在输入样例中，a=4，b=4。根据上面的方法，我们可以求出x = (4(42+1))/(4+42+1) = 16/9。将x代入原来的公式中，得到最小值为(16/9)2+4)0.5 + ((4-16/9)2+1)0.5 ≈ 5。

基于上述解题思路给出的伪代码为：

// 定义一个主函数

main()

{

// 定义两个双精度浮点数变量a和b

double a, b;

// 从标准输入读取a和b的值

input a, b;

// 根据公式计算x的值

double x = (a \* (b \* b + 1)) / (a + b \* b + 1);

// 根据公式计算ans的值

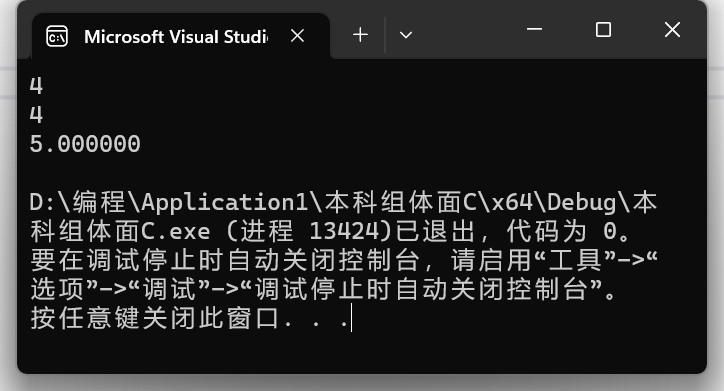
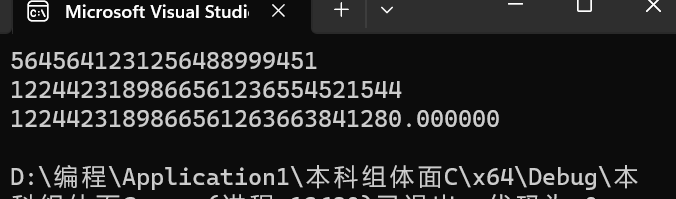
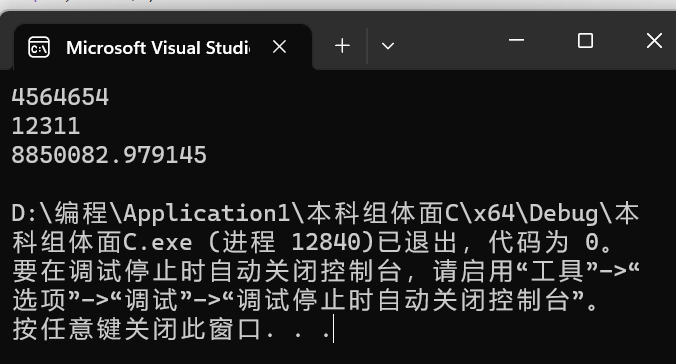
double ans = sqrt(x \* x + a) + sqrt((b - x) \* (b - x) + 1);

// 以六位小数的精度输出ans的值到标准输出

output ans with 6 decimal places;

// 结束主函数

end main;

}测试：【对应本题源程序为:D:\编程\Application1\本科组题面C\FileName.cpp】

**题目F** 太阳能板

题面:

太阳能光发电是指无需通过热过程直接将光能转变为电能的发电方式。它包括光伏发电、光化学发电、光感应发电和光生物发电。光伏发电是利用太阳能级半导体电子器件有效地吸收太阳光辐射能，并使之转变成电能的直接发电方式，是当今太阳光发电的主流。光伏板组件是一种暴露在阳光下便会产生直流电的发电装置，由几乎全部以半导体物料（例如硅）制成的固体光伏电池组成。简单的光伏电池可为手表以及计算机提供能源，较复杂的光伏系统可为房屋提供照明以及交通信号灯和监控系统，并入电网供电。现在，科研人员想要研发新的太阳能板材料，在 n 个仓库中存放了 n 种原始材料，有 n‐1 条道路将这 n 个仓库连接在一起，每条道路连接着两个仓库，进行新材料的合成必须使用两种原始材料。考虑到运输成本和材料成本，只能选择被一条道路直接相连的两个仓库中的原始材料进行合成，且每种原始材料只能被使用一次。在两种原始材料合成之后，得到的新材料的吸光能力为两种原始材料的吸光能力之乘积。现在科研人员想要知道，合成的新材料的吸光能力的总和大是多少。

输入说明：

第一行一个正整数：n

第二行到第 n 行每行两个整数：ai, bi 表示仓库 ai 和仓库 bi 之间存在道路，

第 n+1 行 n 个正整数：vi，表示仓库 i 中的原始材料的吸光能力

输出说明：

能够得到的最大的吸光能力总和。

输入样例：

5

1 2

1 3

2 4

2 5

1 2 3 4 5

输出样例：

13

数据范围：

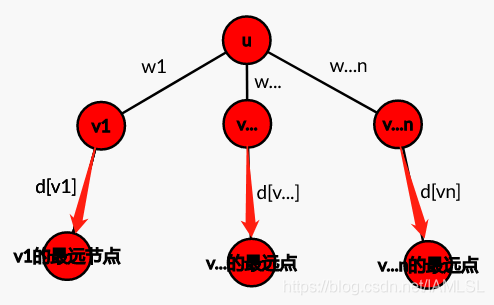
n<=1000000

解题思路:

这个题目是一道典型的树形DP问题，树形DP是一种DP的思想，将DP建立在树状结构的基础上。整体的思路大致就是用树形的结构存储数据。树形DP的关键和实现方法是 dfs。先找到树根，从树根开始运用 dfs 递归，跟 dfs 一样先初始化，然后递归到叶子节点上为止，把最底层的 f[i][j] 更新完毕，再回来往上走，自底向上地根据题意更新上层的 f 数组，

具体操作为: 我们可以使用f[i][0]表示以i为根的子树中，不选i节点的最大吸光能力和，f[i][1]表示以i为根的子树中，选i节点的最大吸光能力和。对于每个节点i，我们枚举它的每个儿子j，然后转移即可。具体来说，如果我们不选i节点，那么我们就需要选它的两个儿子进行合并，因此有f[i][0]=max(f[i][0],f[j][1]+f[k][1])，其中j和k分别是i的两个儿子。如果我们选了i节点，那么我们就不能选它的儿子了，因此有f[i][1]=max(f[i][1],v[i]+max(f[j][0],f[j][1])+max(f[k][0],f[k][1]))。最终答案即为max(f[1][0],f[1][1])。

以下是一个DP树:



伪代码:

// 定义一些常量和变量

MAXN = 10005

n = 树中的节点数

w = 每个节点的权重数组

e = 树中的边数组

head = 每个节点的第一条边数组

cnt = 边数

// 定义一个函数，向树中添加一条边

函数 add(u, v):

// 边数加一

cnt = cnt + 1

// 设置边的目标节点为 v

e[cnt].v = v

// 设置边的下一个指针为 u 的当前头部

e[cnt].next = head[u]

// 设置 u 的头部为新的边

head[u] = cnt

// 定义一个函数，使用动态规划计算每个节点的一些值

函数 dfs(u, fa):

// 为每个节点初始化一个包含四个值的数组 f：

// f[u][0] = 0 (未使用)

// f[u][1] = u 的权重加上所有子节点 v 的 f[v][3] 的和

// f[u][2] = 所有子节点 v 的 max(f[v][2], f[v][3]) 的和

// f[u][3] = 所有子节点 v 的 max(f[v][1], f[v][2], f[v][3]) 的和

f[u][1] = w[u]

f[u][0] = 0

// 遍历 u 的所有边

对于 i = head[u] 到 cnt:

// 获取边的目标节点 v

v = e[i].v

// 如果 v 是 u 的父节点，跳过

如果 v == fa:

继续

// 递归地计算 v 及其子树的 f 值

dfs(v, u)

// 根据公式更新 f[u][1], f[u][2], 和 f[u][3]

f[u][1] = f[u][1] + f[v][3]

f[u][2] = f[u][2] + max(f[v][2], f[v][3])

f[u][3] = f[u][3] + max(f[v][1], max(f[v][2], f[v][3]))

// 主程序

// 从输入读取 n

n = 输入()

// 循环 n-1 次，读取并添加树中的边

对于 i = 1 到 n-1:

// 从输入读取 u 和 v

u, v = 输入()

// 向树中添加 (u, v) 和 (v, u) 这两条边

add(u, v)

add(v, u)

// 循环 n 次，读取并存储每个节点的权重

对于 i = 1 到 n:

// 从输入读取 w[i]

w[i] = 输入()

// 将所有节点和值的 f 初始化为零

f = 零数组，大小为 n x 4

// 在没有父节点的情况下，对根节点调用 dfs 函数

dfs(1, 0)

// 将 f[1][1], f[1][2], 和 f[1][3] 中的最大值加一作为输出打印出来

输出(max(max(f[1][1], f[1][2]), f[1][3]) )

运行结果:【D:\编程\Application1\本科组题面F\FileName.cpp】

