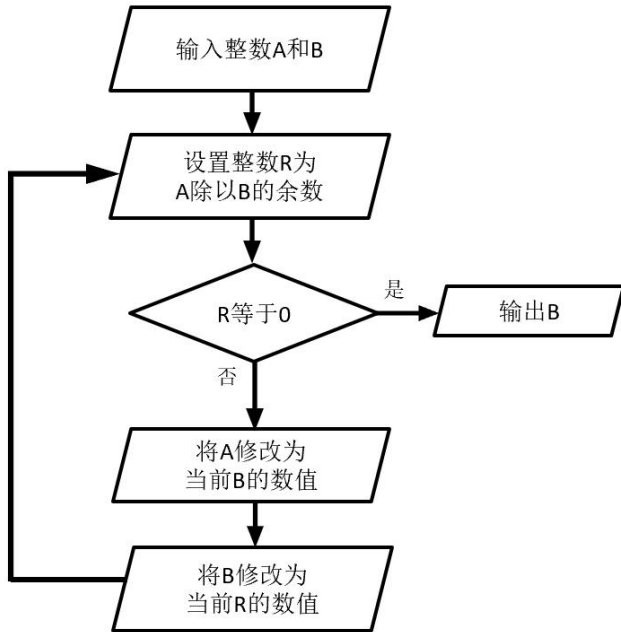


# 2025年“思维100”STEM应用能力科教活动（春季）

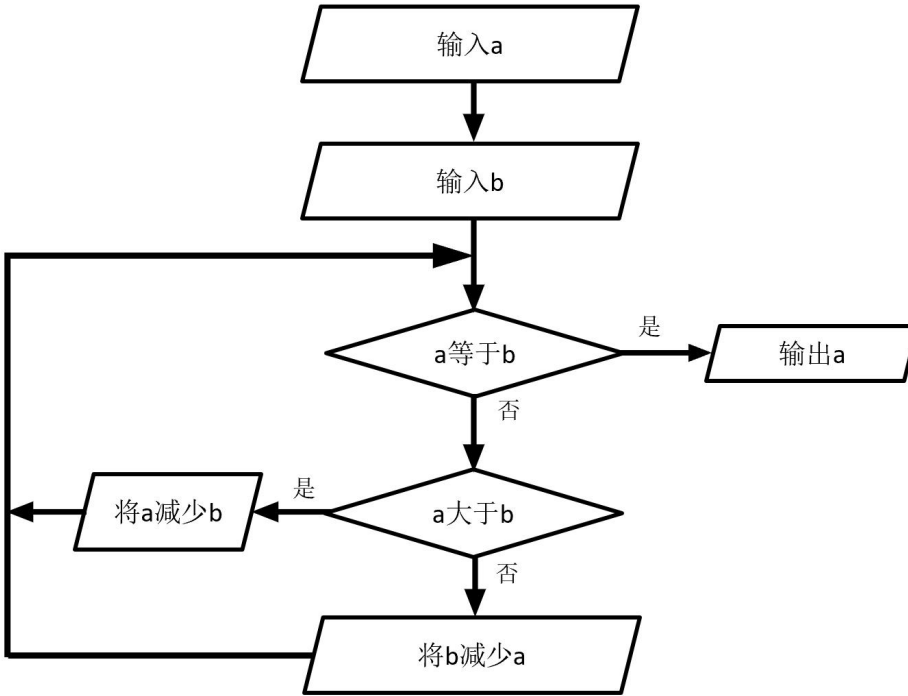
## 五年级参考内容

1. 根据以下流程图，当输入数值  $A=462$ ， $B=2024$  时，输出数值是\_\_\_\_\_。



【答案】22

2. 根据以下流程图，当输入数值  $a=2024$ ， $b=230$  时，输出结果是\_\_\_\_\_。



【答案】46

3. 在科普读物上思思了解到有一种物质，每经过 1 年其质量会减半。如果有 100 克该物质，经过\_\_\_\_\_年后剩余物质就会少于 0.05 克。

【答案】11

4. 有一排植物，它们的初始高度为 1 厘米、3 厘米、5 厘米、7 厘米、9 厘米。有一种营养液可以使植物长高，但每次只能针对当前最矮的一棵植物，使它长高 10 厘米。至少使用\_\_\_\_\_次营养液，才会有一棵植物的高度不低于 2024 厘米。

【答案】1008

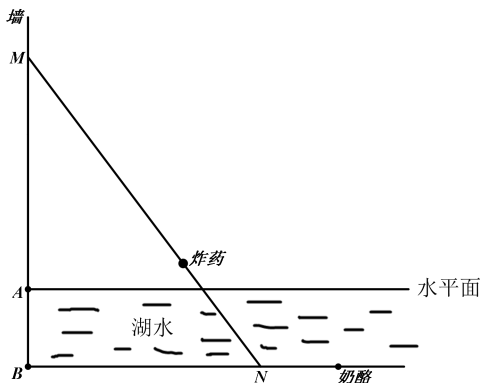
5. 有两个质数  $X$  和  $Y$ ，已知  $X$  小于  $Y$ ，且  $X$  和  $Y$  的总和为 128。符合要求的  $(X, Y)$  共有\_\_\_\_\_种可能。

【答案】3

6. 某个国家的奥运代表队，由运动员和教练员组成。运动员总人数正好和教练员总人数相等，女生人数恰好是总人数的三分之一，男生运动员有 35 人。这个国家的奥运代表队总人数至少有\_\_\_\_\_人。

【答案】72

7. 如下图，一堵墙垂直地竖立在湖边，湖水的深度  $AB$  为 10 分米。有一根木杆  $MN$  的一端靠着墙上，另一端插在湖水中， $MN$  的长度为 50 分米。 $MN$  上靠近点  $N$  的三等分点处绑了一个特殊的炸药，只要被浸没在湖水内部，炸药就会爆炸（在水平面边界上还不会爆炸）。一只水老鼠被绑在木杆的点  $N$  处，它想去吃远处的奶酪，所以它会拉着木杆进行移动。为了能够吃到奶酪，并且不引爆炸药，那么奶酪离开墙的距离最多为\_\_\_\_\_分米。



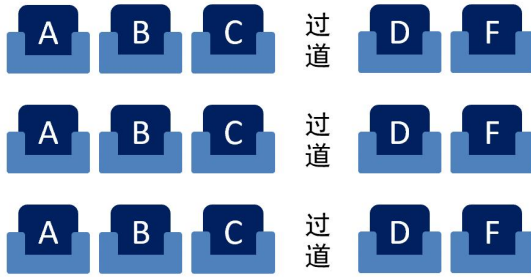
【答案】40

8. 将 5 个红球和 5 个蓝球排成一行，要求不存在三个相邻的同色球，不同的排法有\_\_\_\_\_种。

【答案】84

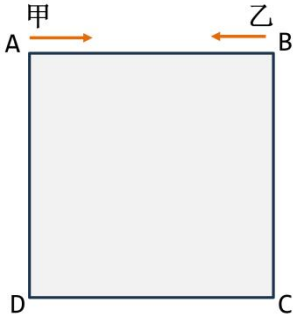
9. 在高铁的线上选座系统里，有 3 排共 15 个座位可以考虑，如图所示。已知有 6 个座位已经被占，那么被占座位有\_\_\_\_\_种可能情况，使得剩余座位里还能选到左右 2

个相邻座位。注意：C 和 D 不算相邻。



【答案】4997

10. 正方形  $ABCD$  边长为 12，甲、乙两人沿着正方形的轮廓匀速行走。甲从  $A$  出发顺时针行走，乙从  $B$  出发逆时针行走，已知甲的速度是乙的 3 倍。他们第 10 次的相遇点为  $X$ ，第 2024 次的相遇点为  $Y$ 。线段  $AY$  和  $BX$  的交点是  $Z$ ，正方形被分成四块区域。这四块区域中，面积最小的区域，其面积是\_\_\_\_\_；面积最大的区域，其面积是\_\_\_\_\_。



【答案】4.5, 85.5

11. 一共有 17 名学生参加了一次测评，测评结束后，老师让这 17 名学生猜测这次测评的平均分。小明的得分是 100 分，他猜测这次测评的平均分是 97 分。剩下的 16 名学生都猜测测评的平均分是  $30 + \frac{n}{2}$  分，其中  $n$  表示这个学生的得分。比如小红的得分是 86 分，她就猜测平均分是  $30 + \frac{86}{2} = 73$  分。小桐的得分是 92 分，她就猜测平均分是  $30 + \frac{92}{2} = 76$  分。若这 17 名学生各自猜测的平均分的平均数，和这次测评的平均分相等，则这次测评的平均分为\_\_\_\_\_分。

【答案】62

12.  $S(N)$  表示正整数  $N$  的数字之和，比如  $S(123) = 1 + 2 + 3 = 6$ 。若  $p$  是三位素数，满足  $S(p) = S(p^2)$ ，则  $p$  的最小值为\_\_\_\_\_。

【答案】199

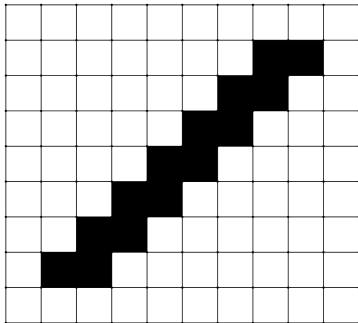
13. 在一款电脑游戏中，共有 8 只怪兽，编号 1~8。两位玩家轮流选择打怪目标，甲先行动。若甲玩家消灭了  $i$  号怪兽，甲会获得  $xi$  枚金币；若乙玩家消灭了  $i$  号怪兽，乙会

获得  $y_i$  枚金币，具体信息如下。两位玩家的目标都是希望游戏结束后自己的金币数减去对方的金币数尽可能大。已知两位玩家都足够聪明，那么最后获得更多金币的是\_\_\_\_\_（填“甲”或“乙”），两人的金币数量相差\_\_\_\_\_枚。

$i$	1	2	3	4	5	6	7	8
$x_i$	12	17	32	24	14	19	28	9
$y_i$	31	20	13	11	19	16	13	26

【答案】甲，7

14. 如图， $10 \times 9$  的方阵中有部分黑色区域，用  $2 \times 1$  的瓷砖（瓷砖可以旋转）去覆盖这个方阵，要求所有瓷砖都正好覆盖  $10 \times 9$  方阵中的两小格，黑色区域不能被覆盖住，而且两块瓷砖不能同时覆盖同一个小方格。最多能放入\_\_\_\_\_块瓷砖。



【答案】36

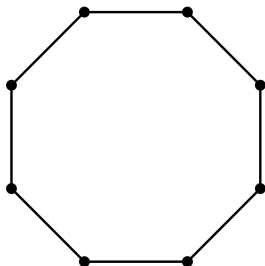
15. 如图，将 1、2、3、4、5、6、7、8 填入正八边形的八个顶点处（每个顶点写一个数），再将 1、2、3、4、5、6、7、8 填入正八边形的八条边上（每个边上写一个数），要求：

(1) 如果一条边的两个端点处所写的数之和为偶数，则这条边上写的数也必须为偶数；

(2) 如果一条边的两个端点处所写的数之和为奇数，则这条边上写的数也必须为奇数。

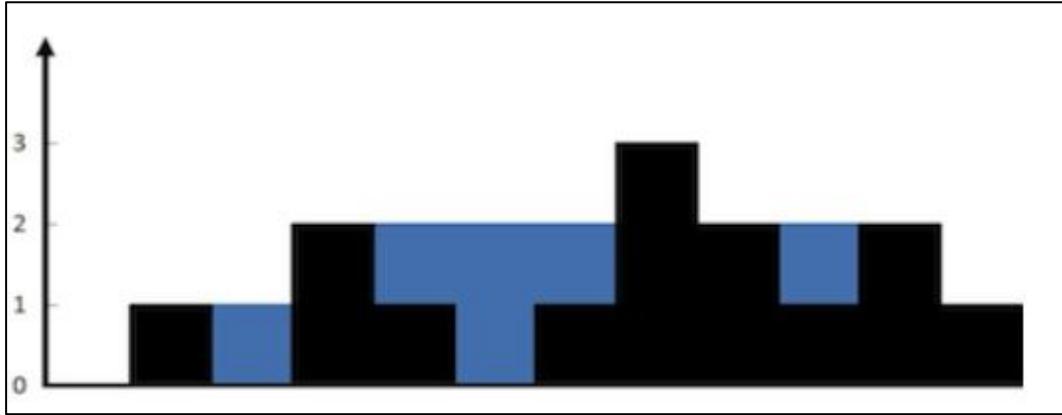
如果两种填法旋转后相同，认为是同一种填法；如果两种填法翻折后相同，认为是不同的填法。

一共有  $N$  种不同的填法，则  $N$  的末两位数字为\_\_\_\_\_。

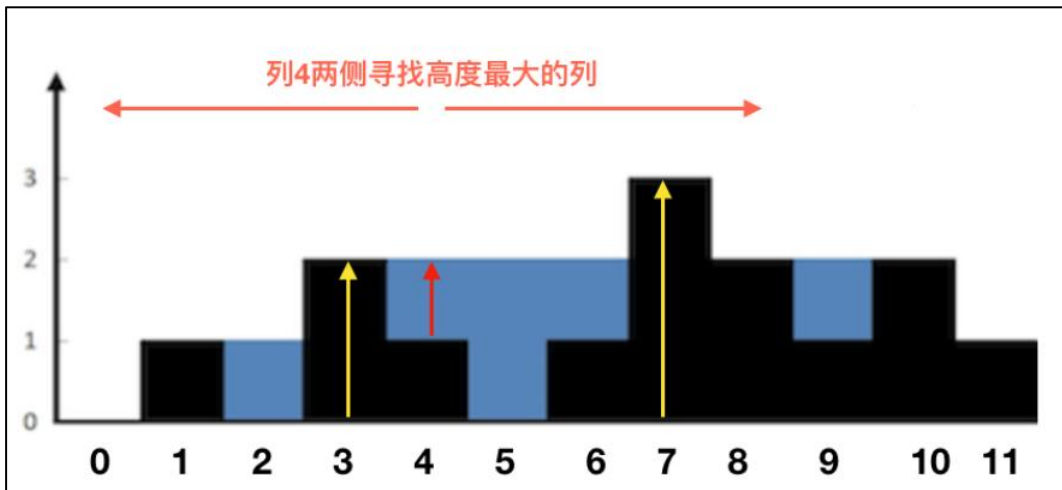


【答案】92

16. 给定一个长度为  $n$  的非负整数数组  $height$ ，其中每个元素表示宽度为 1 的柱子的高度。如下图是由数组  $height=[0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1]$  表示的高度，黑色部分为柱子。在下了足够大的雨之后，柱子与柱子之间将承接一些雨水（蓝色部分表示雨水）。图中的情况下，这些柱子可以接 6 个单位的雨水。



对于给定的数组，如何计算这些柱子排列能够接住多少单位的雨水？最直接的方法是先计算每一列雨水的高度（宽度都为 1），然后把所有雨水列的高度都加起来，就是所有雨水的容量了。那如何计算每一列雨水的高度呢？可以从图中看出，每一列雨水的高度，取决于该列左侧最高的柱子和右侧最高的柱子中，较矮的那个柱子的高度，与该列柱子高度的差。例如下图，我们来计算列 4 雨水的高度。



列 4 左侧最高的柱子是列\_\_\_\_\_，高度为\_\_\_\_\_。  
 列 4 右侧最高的柱子是列\_\_\_\_\_，高度为\_\_\_\_\_。  
 列 4 柱子的高度为\_\_\_\_\_，那么列 4 的雨水体积为\_\_\_\_\_。

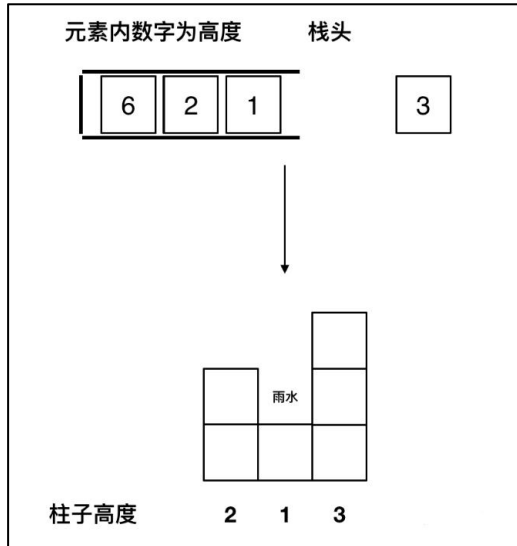
【答案】3, 2, 7, 3, 1, 1

17. 按照上一题的算法，当  $height=[4, 2, 0, 3, 2, 5]$  时，能接住的雨水的体积是\_\_\_\_\_。

【答案】9

18. 上面的暴力算法，随着数组中的元素数量增多，计算会变得复杂。现在我们用单调栈

解法来优化这个算法。单调栈就是不仅要保持栈的特性（先进后出），还要保证栈内元素有序。单调栈通常是一维数组。如下图，只有比栈顶元素小的元素才能直接进栈，否则需要先将栈中比当前元素小的元素出栈，再将当前元素入栈。这样就保证了栈中保留的都是比当前入栈元素大的值，并且从栈顶到栈底的元素值是单调递增的。



从栈头（元素从栈头弹出）到栈底的顺序是从小到大的顺序。其中栈中的数字表示柱子的高度，一旦发现添加的柱子高度大于栈头元素了（ $3 > 1$ ），就表明出现凹槽了，而栈头元素（1）就是凹槽底部的柱子，如图中“雨水”的部分，栈头第二个元素就是凹槽左边的柱子（2），添加的元素（3）就是凹槽右边的柱子。

栈里存放各个柱子的下标 `index`，通过下标即可通过访问数组 `height` 来得到柱子下标对应的高度。当我们遍历每一个柱子时，会出现三种情况：

（1）当前遍历的元素（柱子）高度小于栈顶元素的高度：此时要把这个元素\_\_\_\_\_。

- A. 压入    B. 弹出

（2）当前遍历的元素（柱子）高度等于栈顶元素的高度：此时要更新栈顶元素下标，因为遇到相同高度的柱子，需要使用最右边的柱子来计算宽度。

（3）当前遍历的元素（柱子）高度大于栈顶元素的高度：此时就出现凹槽了。

我们取栈顶元素，即将栈顶元素弹出，这个就是凹槽的底部，也就是中间雨水的位置，对应的下标记为 `mid`，对应的高度为 `height[mid]`（即为上图中的高度 1）。此时的栈顶元素 `st.top()`，就是凹槽的左边位置，下标为 `st.top()`，对应的高度为 `height[st.top()]`。

当前遍历的元素 `i`，就是凹槽右边的位置，下标为 `i`，对应的高度为 `height[i]`（即为上图中的高度 3）。则用 `height[st.top()]`、`height[i]`、`height[mid]` 表示的雨水高度为  $\min(\text{height}[\text{st.top()}], \text{height}[i]) - \text{height}[\text{mid}]$ ，雨水的宽度为： $i - \text{st.top()} - 1$ 。

根据上述单调栈的算法，当输入为  $height=[3, 4, 1, 3, 3, 4, 1, 1, 3, 2, 3, 3]$  时，能接住的雨水的体积是\_\_\_\_\_。

【答案】A, 10

19. “重复的子字符串”问题是指，给定一个非空的字符串，判断它是否可以由它的一个子串重复多次构成。其中给定的字符串只含有小写英文字母，并且长度不超过 10000。下面是一些例子：

示例 1：输入：“abab”，输出：True，解释：可由子字符串“ab”重复两次构成。

示例 2：输入：“aba”，输出：False。

(1) 最暴力简单的方法是可以通过以下几个步骤来判断一个字符串是否可以由其子串重复多次构成。

①观察字符串长度：假设字符串的长度为  $n$ 。如果存在重复的子字符串，则这个子字符串的长度一定是  $n$  的因数。

②寻找因数：找到所有可能的子字符串长度，这些长度是  $n$  的因数。例如：如果  $n$  是 6，则可能的子字符串长度有 1、2、3 和 6。

③构建重复字符串：对于每一个因数长度  $len$ ，提取字符串的前  $len$  个字符作为候选子串。将这个候选子串重复  $(n/len)$  次，构造一个新字符串。

④比较字符串：如果构造的字符串与原字符串相等，则原字符串可以由该子串重复构成，返回 True。如果没有任何候选子串能构成原字符串，则返回 False。

如果现在有一个长度为 8 的字符串，且我们选择的候选子串长度为 4，那么该字符串是候选子串重复\_\_\_\_\_次构造而成。

【答案】2

(2) 这样的算法运行效率太低了，我们考虑更加高效的算法。KMP (Knuth-Morris-Pratt) 算法是一种用于字符串匹配的高效算法，它可以帮助我们判断一个字符串是否包含另一个字符串。下面是 KMP 算法的原理。

①计算最长前缀后缀数组 (LPS 数组)

最长前缀：指的是字符串中，从起始位置到当前字符位置的所有字符中，与当前字符及其之前的部分相同的最大长度的前缀。

最长后缀：指的是字符串中，从当前字符位置到结束位置的所有字符中，与当前字符及其之前的部分相同的最大长度的后缀。

LPS 数组的计算：对于每个字符，LPS 数组中的值表示在该位置的字符串中，最长的可匹配前缀和后缀的长度。可以通过维护一个指针，逐个比较字符，并在匹配时更新 LPS 值。具体步骤请参考下文的示例。

②判断重复子字符串

计算字符串的 LPS 数组后，检查字符串的最后一个字符的 LPS 值 (假设为  $LPS[n-$

1])。如果字符串的长度  $n$  减去  $LPS[n-1]$  的值能整除  $n$ ，则说明字符串可以由重复的子串构成。

我们以字符串“abab”为例，逐步说明 KMP 算法的应用。

①计算 LPS 数组：对于“abab”，我们依次计算每个字符的 LPS 值。

$LPS[0]=0$ ：字符串的第一个字符是“a”，没有任何字符可以构成前缀和后缀，因此没有相同的前缀和后缀，LPS 值为 0。

$LPS[1]=0$ ：字符串的前两个字符是“ab”，目前考虑的前缀是“a”，后缀是“b”，这两个部分没有任何相同的部分，因此 LPS 值仍然为 0。

$LPS[2]=1$ ：字符串的前 3 个字符是“aba”，前缀部分为“a”，后缀部分为“a”（考虑到后缀从字符串开头到当前字符位置的部分），由于它们相同，最长的可匹配前缀和后缀的长度为 1，因此 LPS 值为 1。

$LPS[3]=2$ ：字符串的前 4 个字符是“abab”，前缀部分为“ab”，后缀部分也是“ab”，这两个部分相同，所以最长可匹配前缀和后缀的长度为 2，因此 LPS 值为 2。

于是我们得到 LPS 数组为 [0, 0, 1, 2]。

②判断是否可以由子串重复构成：计算  $n-LPS[n-1]=4-2=2$ ， $4\%2=0$ ，所以可以确认字符串“abab”是由子串“ab”重复构成的。

现在，有一个字符串的长度为 6，且最后一个字符的 LPS 值为 4，那么该字符串是由其子串重复\_\_\_\_\_次构造而成。

**【答案】** 3

(3) 请你使用 KMP 算法计算字符串“ababab”的 LPS 数组：

$LPS[0]=$ \_\_\_\_\_， $LPS[1]=$ \_\_\_\_\_， $LPS[2]=$ \_\_\_\_\_， $LPS[3]=$ \_\_\_\_\_，  
 $LPS[4]=$ \_\_\_\_\_， $LPS[5]=$ \_\_\_\_\_。

**【答案】** 0, 0, 1, 2, 3, 4