

Part C: Prize Questions 1–2

This section has two optional prize questions. They are not part of the core competition and do not contribute to your overall score. You can still get a perfect score without attempting them.

Results for these questions will only be used in determining prize winners. If you would like to be considered for a prize, you are invited to attempt these questions. Prize questions should only be attempted after you have completed your responses for Questions 1–9.

Each prize question has two parts and has the same rules as a core question in the paper.

Note: the full introductory text is copied below but the examples are omitted.

Prize 1. Capri

You are using your goats to clear blackberry thickets. Your goats are contrary creatures and can't be told what to do.

你正在驱赶山羊去清洁黑莓灌木丛，但这些山羊都很叛逆，并不会完全听从你的指令。

1. Goats will only work for whole days.
山羊的工作时间只能按整天计算。
2. If a goat has been assigned to a thicket, it will not permit any other goat, except Capri, help it clear the thicket.
如果某只山羊被分配到某个灌木丛，那么这只山羊不会允许除 Capri 之外的其他山羊来帮它清理这个灌木丛。
3. Each goat can clear an area of 1 GoatRood (GR) of blackberries per day.
每只山羊每天只能清理 1 GR 的黑莓。
4. Capri:
 - i. Capri will not work by himself.
Capri 不会单独工作。
 - ii. The other goats will let Capri join them.
其他山羊会邀请 Capri 一起工作。

iii. If Capri helps another goat, they clear 2 GR of blackberries per day.

如果 Capri 帮助其他山羊一起清洁，那么这两只山羊一天就能清洁 2 GR 的黑莓。

Each of the other goats has been assigned a blackberry thicket. You know the area of each thicket, and therefore how many days one goat would take to clear it. You want to use Capri so that all thickets are cleared in as few days as possible.

除 Capri 之外的每一只山羊都被分配了一片黑莓灌木丛。你了解每片灌木丛的面积，所以可以计算出每只山羊要花多少天的时间才能清理完它们被分配到的灌木丛。现在你想派出 Capri 参加清洁任务，这样一来就可以将所有灌木丛的总清洁时间尽可能地缩短。（下表展示两个分别需要 6 天和 12 天才能清洁完毕的案例）

- A.** How long would it take to clear seven thickets, of areas 87, 78, 72, 67, 63, 59 and 54 GR?
- B.** Capri insists on a day off between thickets.
How long would it take to clear seven thickets, of areas 54, 52, 49, 46, 44, 41 and 38 GR?

Prize 2. QWERTY

Antony the ant is practising his typing skills.

蚂蚁 Antony 正在练习打字。

When he does *not* want to type any letters, he walks gently around the keyboard, stepping from one key to any adjacent key. For example, to get from D to N without typing he could step along the keys in the order $D \rightarrow R \rightarrow T \rightarrow G \rightarrow H \rightarrow N$. There are many quicker ways to do this.

如果他不想输入任何字母，他可以在键盘上缓缓地从某个字母走向其他相邻字母。例如，如果想从 D 到 N 且中间不输入任何字母，他可以按照 $D \rightarrow R \rightarrow T \rightarrow G \rightarrow H \rightarrow N$ 的顺序行走。当然，除此之外还有更快的行走路线。

When he *does* want to type a letter, he must *jump* onto the key. He can do this from one or two steps away.

如果他想输入某个字母，他必须跳向那个字母。Anthony 可以跳向距离自己所在位置一步或两步远的字母。



Antony:

- always starts at the spacebar at the bottom
总是从键盘底部的空格键出发
- can step from the spacebar onto any key on the bottom row *except Z*
可以从空格键走向底部字母行中除 Z 之外的任何一个字母。
- can jump directly from the spacebar to
可以直接从空格键跳向
 - any key in the bottom row
底部字母行中的任何一个字母
 - any key in the middle row *except A or L*
中间字母行中除 A 和 L 外的任何一个字母
- will not return to the spacebar while typing a word
输入过程中，不会返回空格键
- always types as efficiently as possible, by taking the fewest steps between jumps.
总是通过将每次跳跃之间的步数降至最低来追求高效输入。

A. Antony wants to type a 3-letter passcode. His passcode:

- must include the letter A
- must not repeat any letters.

Which passcode has the greatest number of ways to type it as efficiently as possible, and how many ways are there?

Your answer will be the 3-letter passcode followed by the number of ways.

B. Antony's passcode is 3 letters and does not contain any repeated letters. It:

- can be typed as efficiently as possible in exactly 55 ways
- starts with a letter in the top row
- has *exactly* one letter from his name.

What is the passcode?

Your answer will be the 3-letter passcode.

Part A: Questions 1–6

Each question should be answered by a single choice from A to E.

每题有五个选项，考生应从中选出一个正确答案。

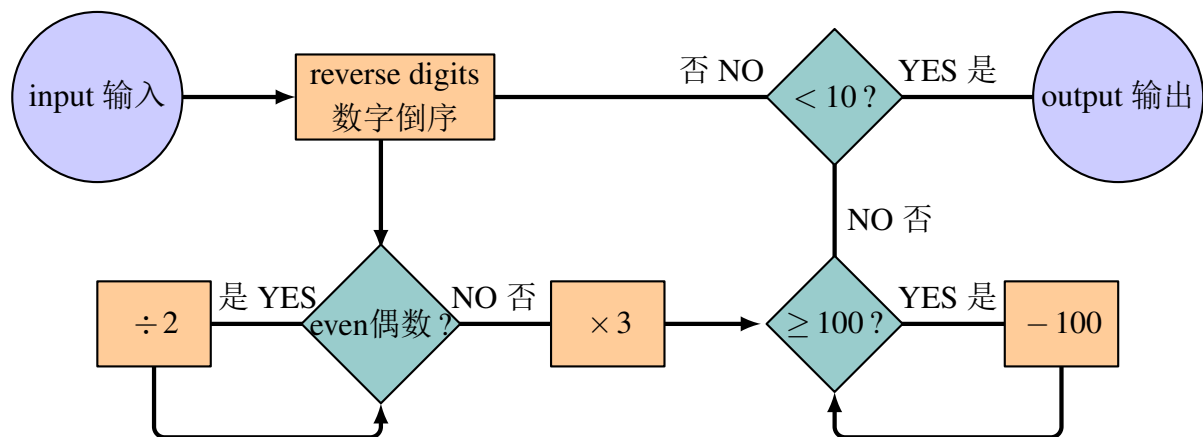
Questions are worth 3 marks each.

每题 3 分。

1. Flow-流程图

The number 38 is input into the following flow chart:

将数字 38 输入下列流程图：



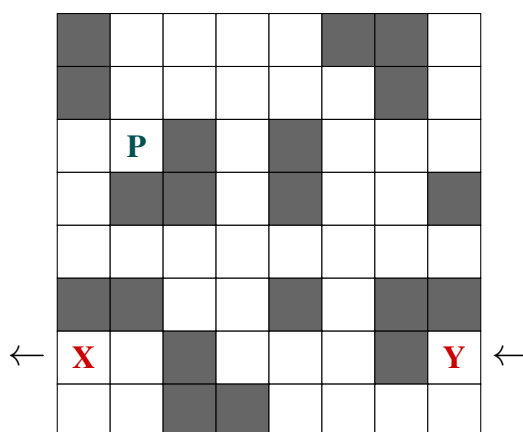
2. Donut Prince-甜甜圈王子

The game *Donut Prince* is played on the 8×8 grid shown. The prince can move one square up, down, left or right on each turn. He cannot move onto a solid grey square and diagonal moves are not allowed.

甜甜圈王子 游戏采用如下的 8×8 网格。王子每轮都可以在上、下、左、右四个方向上移动一个方格。但是，他不能进入灰色方格，也不能在对角线方向上移动。

When the prince is at any of the four edges of the grid, one more move towards that edge transports him to the same position on the opposite side. For example, he can get from square **X** to square **Y** with one move left.

当王子位于该网格四条边中任意一边时，朝着所在边方向再前进一步可以被传送到对边的同一位置。例如，王子可以从 **X** 方格向上移动一步到达 **Y** 方格。



The prince starts at square **P**.

现在王子从 **P** 方格出发。

How many of the white squares could *not* be reached by the prince in 6 moves or fewer?

请问经过 6 次或者更少次数的移动后，王子 并未 涉足的白色方格有多少个？

- (A) 3 (B) 4 (C) 5 (D) 6 (E) 7

3. Row delete–消消乐

In a spreadsheet, rows are labelled 1, 2, 3, ... and columns are labelled A, B, C, ...

在下列图表中，行标记为 1、2、3、...；列标记为 A、B、C、...

When a row is deleted, all of the rows below it shuffle up one position. For example, deleting row 3 means the *new* row 3 is the *old* row 4, and so on.

每消除一行，此行下方的所有行上移一行。例如，消除第 3 行时，原来的第 4 行就会成为新的第 3 行，以此类推。

| | A | B | C | |
|---|---------|---------|---|--|
| 1 | Antares | Scorpio | M | |
| 2 | Rigel | Orion | B | |
| 3 | Pollux | Gemini | K | |
| 4 | Canopus | Carina | A | |
| 5 | Acrux | Crux | B | |

delete 消除

→

row 3 第 3 行

| | A | B | C | |
|---|---------|----------|---|--|
| 1 | Antares | Scorpio | M | |
| 2 | Rigel | Orion | B | |
| 3 | Canopus | Carina | A | |
| 4 | Acrux | Crux | B | |
| 5 | Polaris | Ursa Min | F | |

Rows are deleted one at a time by a sequence of row numbers such as 3, 5, 2, ...

给定一个行号序列，例如 3、5、2、...，按序列中的顺序逐一消除各行。

This means delete row 3, then delete the *new* row 5 (which is the original row 6), then delete the *new* row 2 (which is the original row 2), and so on.

首先消除第 3 行，再消除新的第 5 行（即原先的第 6 行），之后消除新的第 2 行（即原先的第 2 行），以此类推。

Different sequences may or may not have different effects. For example, deleting 3, 1, 3 has the same effect as deleting 5, 1, 2. Both of these delete, in some order, the rows that were originally numbered 1, 3 and 5.

不同序列可能会也可能不会产生不同效果。例如，按照 3, 1, 3 的顺序消除行的效果与按 5, 1, 2 的顺序消除行的效果相同。以上两种都按某种顺序消除了原先的 1、3、5 行。

The cost of a sequence is the sum of the row numbers: 3, 1, 3 has cost $3 + 1 + 3 = 7$ and 5, 1, 2 has cost $5 + 1 + 2 = 8$. So 3, 1, 3 is cheaper than 5, 1, 2.

序列成本即序列中行号总和，例如序列 3, 1, 3 的成本为 $3 + 1 + 3 = 7$ ；序列 5, 1, 2 的成本为 $5 + 1 + 2 = 8$ 。因此，3, 1, 3 的成本比 5, 1, 2 的成本更低。

What is the cost of the cheapest sequence that has the same effect as 3, 1, 4, 1, 5, 9, 2, 6, 5?
请问与序列 3, 1, 4, 1, 5, 9, 2, 6, 5 效果相同的序列中，最便宜序列的成本是多少？

(A) 21

(B) 23

(C) 25

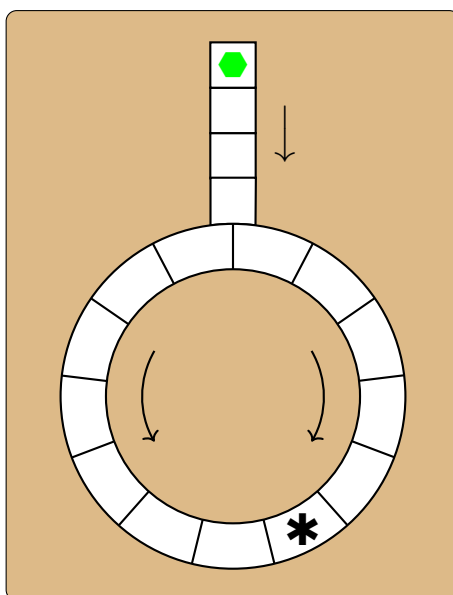
(D) 27

(E) 29

4. Your turn first-先发制人

Your friend has challenged you to a game he has invented. It starts with a counter ◆ on the topmost cell. Players take it in turns to move the counter 1, 2 or 3 cells in the direction of the arrow. An interesting feature of the game is that there is a choice of moving the counter anti-clockwise or clockwise when it reaches the circle. Once the counter is moved clockwise or anti-clockwise, all subsequent moves must be in that direction. The winner is the player who moves the counter onto the cell with the *****.

你的朋友在他设计的一款游戏中向你发起挑战。游戏规则如下：首先用一个筹码 ◆ 标记玩家位置。玩家从顶部方格开始，轮流按照箭头方向将 ◆ 移动1、2或3格。该游戏有一个非常有意思的特点：玩家抵达圆圈时可以选择顺时针方向移动或逆时针方向移动。玩家一旦选择顺时针或逆时针方向，此后只能按照这个方向在圆圈中移动。将 ◆ 移动到带有 ***** 的方格中的玩家获胜。



Your friend is a very skilled player. You have the first move. The *only* way for you to guarantee a win is by

你的朋友是一位经验非常丰富的玩家。游戏开局时，你先手，此时唯一能确保你获胜的方法是

- (A) moving the counter 1 cell 将筹码移动 1 格
- (B) moving the counter 2 cells 将筹码移动 2 格
- (C) moving the counter 3 cells 将筹码移动 3 格
- (D) moving the counter 1 cell or 3 cells 将筹码移动 1 格或 3 格
- (E) moving the counter 2 cells or 3 cells 将筹码移动 2 格或 3 格

5. Mary-程序员玛丽

Mary has written a program to generate random sequences of the letters of her name.

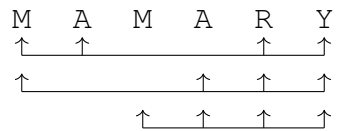
Mary 设计了一款用她名字中的字母来生成随机字母序列的程序。

She is interested in how many times the letters M A R Y, in order, can be extracted from a sequence.

她很好奇到底有多少种方法可以从该程序生成的某个序列中按顺序提取出字母 M A R Y。

For instance M A R Y can be extracted from the sequence M A M A R Y in 3 ways.

例如，有三种方法可以从序列 M A M A R Y 中按顺序提取出字母 M A R Y。



Mary's program generates the sequence M R A Y R Y M A R Y A M Y R Y.

Mary 的程序可以生成序列 M R A Y R Y M A R Y A M Y R Y。

In how many ways can M A R Y be extracted from this sequence?

请问有多少种方法可以在该序列中按顺序提取字母 M A R Y?

(A) 10

(B) 12

(C) 14

(D) 16

(E) 18

6. Card choices-抽卡

Tyson is playing a card game. He has six pairs of numbered cards, all visible, as follows:

Tyson 正在玩一种卡牌游戏。他有六组可见卡牌，如图所示：



Tyson can choose *at most one* card from each pair, and aims to make the highest possible total.

Tyson 从每组卡牌中最多抽取一张卡牌，使得抽取的卡牌总和尽可能达到最大值。

However, as he moves from left to right, any card he chooses must be higher than the card previously chosen.

但是，当从左向右抽取卡牌时，他抽取的任何一张卡牌上的数字必须比前一张抽取的卡牌上的数字大。

Examples 举例

Valid sequence 有效序列: 5 _ 9 _ 10 _

Invalid sequence 无效序列: 1 5 3 _ _ _

What is the highest total Tyson can make?

Tyson 抽取的卡牌上的数字总和最大能达到多少？

(A) 24

(B) 25

(C) 26

(D) 27

(E) 28

Part B: Questions 7–9

Each question has three parts, each of which is worth 2 marks.

每题有三个部分，每部分 2 分。

Each part should be answered by a number in the range 0–999.

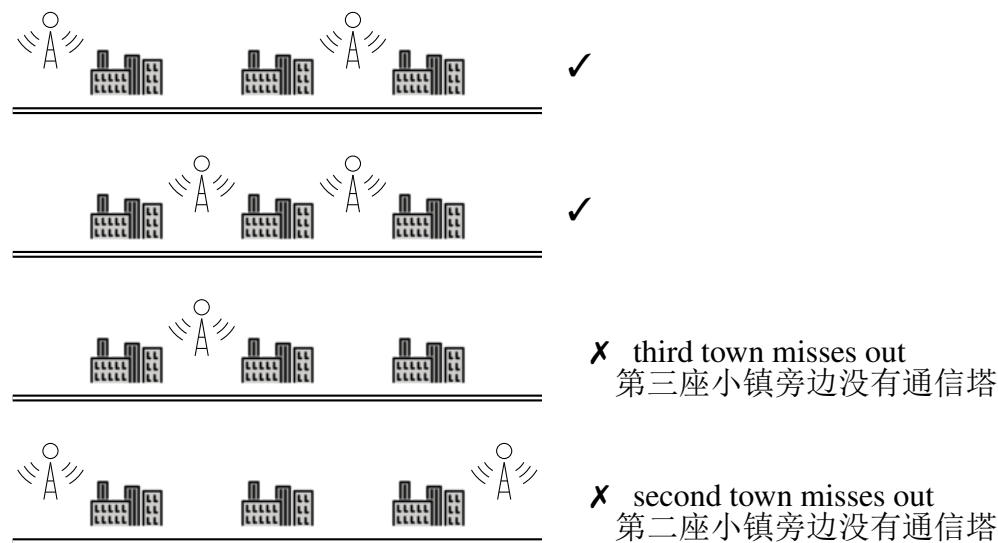
每部分答案应为一个介于 0–999 之间的数字。

7. Communication towers-通信塔

There are several towns along a road. Sites have been identified for communication towers to service the towns. Each town must be in sight of one or two communication towers.

沿路有几座通信塔。搭建通信塔是为了给城市带来更好的通信保障，因此必须确保每个小镇旁边有一座或两座通信塔。

Examples 举例:



You know the cost of building a communication tower on each site. You want the total cost of building the communication towers to be as low as possible.

基于你所掌握的在各个位置搭建通信塔的成本，现在你想将所有通信塔的总体搭建成本尽可能降到最低。

For each of the following, what is the lowest cost to build communication towers so that every town is next to at least one tower?

对于下列每一种通信塔搭建方案，计算可以使得每个小镇旁边都至少有一座通信塔的最低成本。

(Each number represents the cost of building a tower on that site. The Ts represent towns.) (其中各个数字代表在该位置搭建一座通信塔的成本。T 代表小镇。)

A. 4 6 7 4 7 6 4 8 9 6
T T T T T T T T T

B. 4 7 6 6 9 7 6 4 2 4 1
T T T T T T T T T T

C. 3 4 5 3 2 3 1 4 3 4 6 5 4
T T T T T T T T T T T T

8. Capri 山羊

You are using your goats to clear blackberry thickets. Your goats are contrary creatures and can't be told what to do.

你正在驱赶山羊去清洁黑莓灌木丛，但这些山羊都很叛逆，并不会完全听从你的指令。

1. Goats will only work for whole days.
山羊的工作时间只能按整天计算。
2. If a goat has been assigned to a thicket, it will not permit any other goat, except Capri, help it clear the thicket.
如果某只山羊被分配到某个灌木丛，那么这只山羊不会允许除 Capri 之外的其他山羊来帮它清理这个灌木丛。
3. Each goat can clear an area of 1 GoatRood (GR) of blackberries per day.
每只山羊每天只能清理 1 GR 的黑莓。
4. Capri:
 - i. Capri will not work by himself.
Capri 不会单独工作。
 - ii. The other goats will let Capri join them.
其他山羊会邀请 Capri 一起工作。
 - iii. If Capri helps another goat, they clear 2 GR of blackberries per day.
如果 Capri 帮助其他山羊一起清洁，那么这两只山羊一天就能清洁 2 GR 的黑莓。

Each of the other goats has been assigned a blackberry thicket. You know the area of each thicket, and therefore how many days one goat would take to clear it. You want to use Capri so that all thickets are cleared in as few days as possible (6 days and 12 days in the examples below).

除 Capri 之外的每一只山羊都被分配了一片黑莓灌木丛。你了解每片灌木丛的面积，所以可以计算出每只山羊要花多少天的时间才能清理完它们被分配到的灌木丛。现在你想派出 Capri 参加清洁任务，这样一来就可以将所有灌木丛的总清洁时间尽可能地缩短。（下表展示两个分别需要 6 天和 12 天才能清洁完毕的案例）

Examples (Rem. is the area remaining to be cleared.)

举例 (Rem.指的是剩余待清洁区域.)

Two thickets, of areas 11 and 4 GR.

两片灌木丛，面积分别为 11 和 4 GR。

Capri (*) helps clear thicket 1 for the first 5 days.

Capri(*) 在前 5 天协助清洁 1 号灌木丛。

English version 英文版本:

| Days | Thicket 1 (11) | | Thicket 2 (4) | |
|------|----------------|------|---------------|------|
| | Cleared | Rem. | Cleared | Rem. |
| 1-4 | 8* | 3 | 4 | 0 |
| 5 | 2* | 1 | | |
| 6 | 1 | 0 | | |

Chinese version 中文版本:

| 天数 | 1 号灌木丛(11) | | 2 号灌木丛(4) | |
|-----|------------|------|-----------|------|
| | 已清洁 | 待清洁. | 已清洁 | 待清洁. |
| 1-4 | 8* | 3 | 4 | 0 |
| 5 | 2* | 1 | | |
| 6 | 1 | 0 | | |

6 days are required to clear the thickets.

所有灌木丛清洁完毕需要 6 天。

Two thickets, of areas 20 and 15 GR.

两片灌木丛，面积分别为 20 和 15 GR。

Capri (*) helps clear thicket 1 for 8 days then thicket 2 for 3 days.

Capri(*) 用了 8 天协助清洁 1 号灌木丛，然后又用了 3 天协助清洁 2 号灌木丛。

English version 英文版本:

| Day | Thicket 1 (20) | | Thicket 2 (15) | |
|------|----------------|------|----------------|------|
| | Cleared | Rem. | Cleared | Rem. |
| 1-8 | 16* | 4 | 8 | 7 |
| 9-11 | 3 | 1 | 6* | 1 |
| 12 | 1 | 0 | 1 | 0 |

Chinese version 中文版本:

| 天数 | 1 号灌木丛(20) | | 2 号灌木丛(15) | |
|------|------------|-----|------------|-----|
| | 已清洁 | 待清洁 | 已清洁 | 待清洁 |
| 1-8 | 16* | 4 | 8 | 7 |
| 9-11 | 3 | 1 | 6* | 1 |
| 12 | 1 | 0 | 1 | 0 |

12 days are required to clear the thickets.

所有灌木丛清洁完毕需要 12 天。

(Two tables in each example are same, but in different languages)

(同一案例中的两个表格内容相同，仅为方便不同语言同学使用)

For each of the following, find the minimum number of days required to clear all the thickets.

求下列各项中所有灌木丛清洁完毕所需的最短天数。

A. Three thickets, of areas 40, 22 and 16 GR.

三片灌木丛，面积分别为 40、22、16 GR

B. Four thickets, of areas 47, 37, 27 and 17 GR.

四片灌木丛，面积分别为 47、37、27、17 GR

C. Four thickets, of areas 40, 35, 30 and 25 GR.

四片灌木丛，面积分别为 40、35、30、25 GR

9. QWERTY-蚂蚁打字

Antony the ant is practising his typing skills.

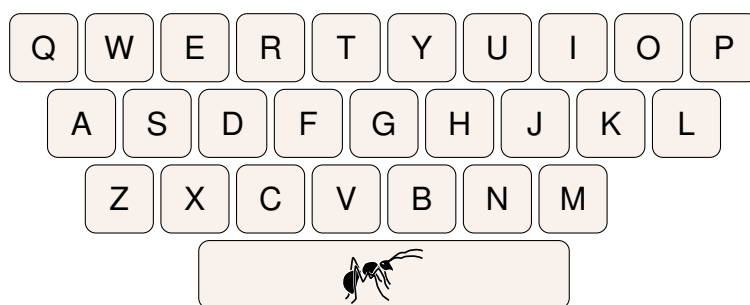
蚂蚁 Antony 正在练习打字。

When he does *not* want to type any letters, he walks gently around the keyboard, stepping from one key to any adjacent key. For example, to get from D to N without typing he could step along the keys in the order $D \rightarrow R \rightarrow T \rightarrow G \rightarrow H \rightarrow N$. There are many quicker ways to do this.

如果他不想输入任何字母，他可以在键盘上缓缓地从某个字母走向其他相邻字母。例如，如果想从 D 到 N 且中间不输入任何字母，他可以按照 $D \rightarrow R \rightarrow T \rightarrow G \rightarrow H \rightarrow N$ 的顺序行走。当然，除此之外还有更快的行走路线。

When he *does* want to type a letter, he must *jump* onto the key. He can do this from one or two steps away.

如果他想输入某个字母，他必须跳向那个字母。Anthony 可以跳向距离自己所在位置一步或两步远的字母。



Antony:

- always starts at the spacebar at the bottom
总是从键盘底部的空格键出发
- can step from the spacebar onto any key on the bottom row *except Z*
可以从空格键走向底部字母行中除 Z 之外的任何一个字母。
- can jump directly from the spacebar to
可以直接从空格键跳向
 - any key in the bottom row
底部字母行中的任何一个字母
 - any key in the middle row *except A or L*
中间字母行中除 A 和 L 外的任何一个字母
- will not return to the spacebar while typing a word
输入过程中，不会返回空格键
- always types as efficiently as possible, by taking the fewest steps between jumps.
总是通过将每次跳跃之间的步数降至最低来追求高效输入。

Example: To type 'REACH' he could:

举例: 若想输入'REACH', 他可以:

1. step onto C, then jump onto R, E and A
走到 C, 然后跳向 R, E 和 A
2. step onto Z, then jump onto C
走到 Z, 然后跳向 C
3. step onto V, then jump onto H.
走到 V, 然后跳向 H.

This takes 3 steps, shown underlined, and this is the fewest possible. There are other ways he can type 'REACH' with 3 steps. (There are 5 letters so there will be 5 jumps.)

上述步骤中, 蚂蚁 Anthony 在跳跃之间需要走三步, 即走向用下横线突出的三个字母, 这是可能的最少步数。当然也有其他类似的只需要走三步就可以输入'REACH'的路线。(上述有 5 个需要输入的字母所以需要跳跃 5 次。)

For each of the following, find the number of ways that Antony can type the word as efficiently as possible:

请问 Anthony 有多少种路线可以尽可能高效地输入下列各个单词?

A. TYPEWRITER

B. ALGORITHM

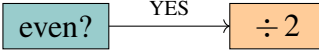
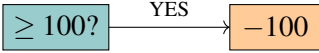
C. COMPUTER

Solutions

Part A: Questions 1–6

1. Flow

There are two key shortcuts:

- the  loop:
this finds the largest odd factor of a number.
- the  loop:
this finds the last two digits of a number (mod 100).

Tracing through the flow chart we get the sequence of numbers in the following table, reading top to bottom then left to right:

| | | | | |
|------------|-----|-----|----|----|
| Input | 38 | | | |
| Reverse | 83 | 94 | 14 | 12 |
| Odd factor | 83 | 47 | 7 | 3 |
| Triple | 249 | 141 | 21 | 9 |
| Mod 100 | 49 | 41 | 21 | 9 |
| Output | 9 | | | |

So the ‘reverse the digits’ process is applied 4 times in all.

Hence (D).

2. Donut Prince

We label the squares that Prince can get to with 1 move, then the squares he could get to in 2 moves, and so on.

| | | | | | | | | | |
|---|---|----------|---|---|---|---|---|---|---|
| | | ↑ | | ↑ | | ↑ | | | |
| | | | 2 | 3 | 4 | 5 | | | |
| | | | 1 | 2 | 3 | 4 | 5 | | |
| ← | 1 | P | | 4 | | 4 | 3 | 2 | ← |
| | 2 | | | 5 | | 5 | 4 | | |
| ← | 3 | 4 | 5 | 6 | | 6 | 5 | 4 | ← |
| | | | 6 | | | | | | |
| ← | 5 | 4 | | | | | | 6 | ← |
| | 4 | 3 | | | 6 | | 6 | 5 | |
| | | ↑ | | ↑ | | ↑ | | | |

There are 7 white squares that the prince could not reach.

Hence (E).

3. Row delete

Solution 1

We first find out the effect of the sequence 3, 1, 4, 1, 5, 9, 2, 6, 5.

Start by listing the row numbers 1, 2, 3, ... in order. The first 3 in the sequence means cross out the 3rd number, which is 3. The 1 means cross out the first number, which is 1. The 4 means cross out the 4th remaining number, which at this point is 6. So after three steps we have the following:

~~1~~ 2 ~~3~~ 4 5 ~~6~~ 7 ...

For each new row number n , cross out the n th remaining number in the list. If at any point additional numbers are needed, just add them to the end. After the full sequence we are left with the following:

~~1~~ ~~2~~ ~~3~~ 4 ~~5~~ ~~6~~ 7 8 ~~9~~ 10 ~~11~~ ~~12~~ 13 ~~14~~ 15 ...

So we need to find the cheapest sequence which ultimately deletes the same collection of rows.

Starting at the left end, we can delete rows 1, 2 and 3 with 1, 1, 1 and clearly this is the cheapest strategy that does this:

~~1~~ ~~2~~ ~~3~~ 4 5 6 7 ...

In order to leave 4 and 7 alone and cross out 5 and 6, we proceed with 2, 2:

~~1~~ ~~2~~ ~~3~~ 4 ~~5~~ ~~6~~ 7 ...

Continuing in this way, we find that the cheapest sequence is 1, 1, 1, 2, 2, 4, 5, 5, 6 which has cost

$$1 + 1 + 1 + 2 + 2 + 4 + 5 + 5 + 6 = 27.$$

Hence (D).

Solution 2

We have to delete the rows originally numbered 1, 2, 3, 5, 6, 9, 11, 12 and 14.

Clearly we will delete them in that order.

The 1 costs 1.

It will cost $2 - 1 = 1$ to delete the 2.

It will cost $3 - 2 = 1$ to delete the 3.

It will cost $5 - 3 = 2$ to delete the 5.

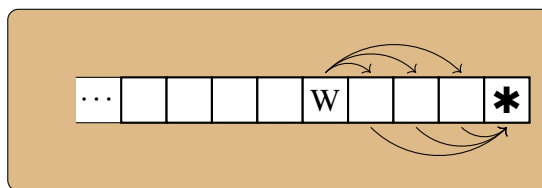
Continuing in this way we have

$$\begin{aligned} &1 + (2 - 1) + (3 - 2) + (5 - 3) + (6 - 4) + (9 - 5) + (11 - 6) + (12 - 7) + (14 - 8) \\ &= (1 + 2 + 3 + 5 + 6 + 9 + 11 + 12 + 14) - (1 + 2 + 3 + 4 + 5 + 6 + 7 + 8) \\ &= (1 + 2 + 3 + 5 + 6 + 9 + 11 + 12 + 14) - ((8 \times 9) \div 2) \\ &= 63 - 36 = 27 \end{aligned}$$

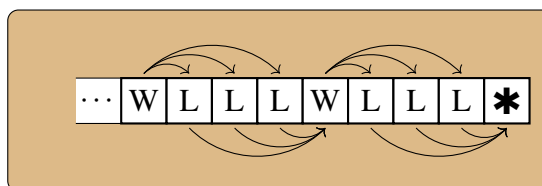
Hence (D).

4. Your turn first

A cell 4 away from the target cell is a winning cell (W), for if you move the counter to the cell you can win on your next move whatever your opponent does.

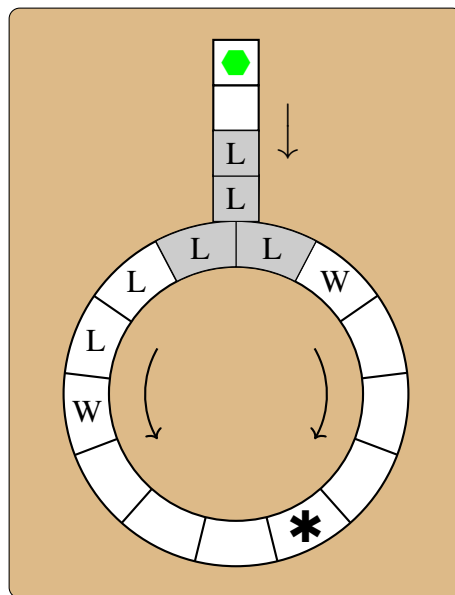


This extends to a cell 4 away from a winning cell is also a winning cell. Cells that are 1, 2 or 3 moves from a winning cell are losing cells.



With this understanding, the aim of the game becomes giving your opponent no choice but to land on a losing square.

For the board, the winning cells on the circle are shown by a W, and the cells within 3 moves are of either W are indicated have an L.



From this diagram we see that if our first move is 2 or 3 we move to a losing cell. However if we move 1 cell our opponent can only move to one of the shaded cells, all of which are losing cells.

Hence (A).

5. Mary

We build a table a row at a time. There will be a row for each of M A R Y. The first row is the M row. In each M column will be a 1.

| | M | R | A | Y | R | Y | M | A | R | Y | A | M | Y | R | Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | 1 | | | | | | 1 | | | | | 1 | | | |

The number in the A column of the second row will be the number of M A sequences finishing at that A. For example there is one M before the first A, but two before each of the others.

| | M | R | A | Y | R | Y | M | A | R | Y | A | M | Y | R | Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | 1 | | | | | | 1 | | | | | 1 | | | |
| A | | | 1 | | | | | 2 | | | 2 | | | | |

We continue with the R row. The R column contains the number of M A R sequences finishing at that R.

| | M | R | A | Y | R | Y | M | A | R | Y | A | M | Y | R | Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | 1 | | | | | | 1 | | | | | 1 | | | |
| A | | | 1 | | | | | 2 | | | 2 | | | | |
| R | | 0 | | | 1 | | | | 3 | | | | | | 5 |

And finally the Y row.

| | M | R | A | Y | R | Y | M | A | R | Y | A | M | Y | R | Y |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M | 1 | | | | | | 1 | | | | | 1 | | | |
| A | | | 1 | | | | | 2 | | | 2 | | | | |
| R | | 0 | | | 1 | | | | 3 | | | | | | 5 |
| Y | | | | 0 | | 1 | | | | 4 | | | 4 | | 9 |

M A R Y can be extracted in $1 + 4 + 4 + 9 = 18$ ways.

Hence (E).

6. Card choices

We will use the notation $n(i)$ for the card numbers, and $t(i)$ for the largest total of the first i numbers subject to the given rules.

We will write the card numbers in a line, with a | to indicate the pairs.

5 1 | 4 5 | 3 9 | 5 7 | 7 10 | 8 9

Then $t(i) = n(i) + t(j)$ where $t(j)$ is the largest t value for all $j < i$. So for our data, $t(1) = n(1) = 5$, $t(2) = n(2) = 1$. Then $t(3) = n(3) + t(2) = 4 + 1 = 5$, etc.

| | | | | | | | | | | | | |
|--------|---|---|---|---|---|----|----|----|----|----|----|----|
| $n(i)$ | 5 | 1 | 4 | 5 | 3 | 9 | 5 | 7 | 7 | 10 | 8 | 9 |
| $t(i)$ | 5 | 1 | 5 | 6 | 4 | 15 | 10 | 13 | 17 | 25 | 25 | 26 |

The largest total following the rules is 26 ($1 + 4 + 5 + 7 + 9$), by choosing the numbers

circled below.

| | | | | | |
|---|---|---|---|----|---|
| 5 | ④ | 3 | ⑤ | ⑦ | 8 |
| ① | 5 | 9 | 7 | 10 | ⑨ |

Hence (C).

Part B: Questions 7–9

7. Communication towers

From the third site onward, if a tower is built on the site, a tower must also be built on exactly one of the two sites to its left.



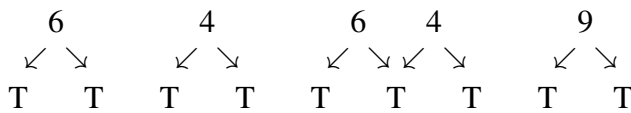
This leads to a left-to-right algorithm where we add sites one at a time, computing the lowest cost so far if a tower was built at that site. This is the cost of building on that site + the lower of the costs of building at the preceding two sites.

Then the lowest cost will be the lower of the costs at the last two sites.

A. 4 6 7 4 7 6 4 8 9 6

| | | | | | | | | | | |
|--------|---|---|----|----|----|----|----|----|----|----|
| cost | 4 | ⑥ | 7 | ④ | 7 | ⑥ | ④ | 8 | ⑨ | 6 |
| lowest | 4 | 6 | 11 | 10 | 17 | 16 | 20 | 24 | 29 | 30 |

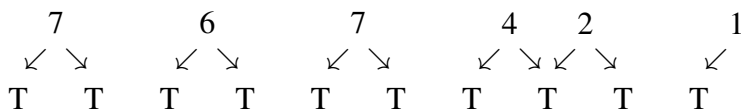
The total cost is $6 + 4 + 6 + 4 + 9 = 29$.



B. 4 7 6 6 9 7 6 4 2 4 1

| | | | | | | | | | | | |
|--------|---|---|----|----|----|----|----|----|----|----|----|
| cost | 4 | ⑦ | 6 | ⑥ | 9 | ⑦ | 6 | ④ | ② | 4 | ① |
| lowest | 4 | 7 | 10 | 13 | 19 | 20 | 25 | 24 | 26 | 28 | 27 |

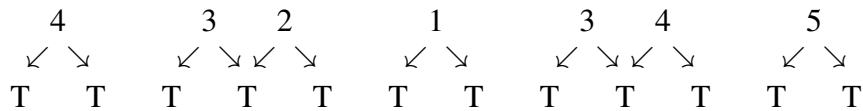
The total cost is $6 + 4 + 6 + 4 + 7 = 27$.



C. 3 4 5 3 2 3 1 4 3 4 6 5 4

| | | | | | | | | | | | | | |
|--------|---|---|---|---|---|----|----|----|----|----|----|----|----|
| cost | 3 | ④ | 5 | ③ | ② | 3 | ① | 4 | ③ | ④ | 6 | ⑤ | 4 |
| lowest | 3 | 4 | 8 | 7 | 9 | 10 | 10 | 14 | 13 | 17 | 19 | 22 | 23 |

The total cost is $4 + 3 + 2 + 1 + 3 + 4 + 5 = 22$.



8. Capri

We will use $t_1 \leq t_2 \dots$ for the number of days to clear thicket 1, thicket 2, ...

Solution 1

Two thickets

Capri only helps clear thicket 1.

This is the case where $t_2 \leq t_1/2$

Capri helps clear both thickets.

This is the case where $t_2 > t_1/2$

- Capri will help with thicket 1 until both thickets have the same time remaining.
- Then he will help clear both thickets equally.

Note that once both thickets have 2 days left, it does not reduce the time to clear both if Capri helps with one of them.

Three thickets

Capri only helps clear thicket 1.

This is the case where $t_2 \leq t_1/2$

Capri helps clear thickets 1 and 2.

This is the case where $t_2 > t_1/2$ and $t_3 \leq t_1/2$.

- Capri will help with thicket 1 until both thickets have the same time remaining.
- Then he will help clear both thickets equally.

Capri helps clear all three thickets.

This is the case where $t_3 > t_1/2$

- Capri will help with thicket 1 until thickets 1 and 2 have the same time remaining.

- Then he will help clear thickets 1 and 2 equally until they have the same time remaining as thicket 3.
- Then he will help clear all thickets equally.

Note that once all the thickets have 2 or 3 days left, it does not reduce the time to clear them all if Capri helps.

We can now solve parts A.

| A. | Thicket 1 (40) | | Thicket 2 (22) | | Thicket 3 (16) | | |
|----|----------------|---------|----------------|---------|----------------|---------|------|
| | Day | Cleared | Rem. | Cleared | Rem. | Cleared | Rem. |
| | 1-16 | 32* | 8 | 16 | 6 | 16 | 0 |
| | 17-18 | 4* | 4 | 2 | 4 | | |
| | 19 | 2* | 2 | 1 | 3 | | |
| | 20 | 1 | 1 | 2* | 1 | | |
| | 21 | 1 | 0 | 1 | 0 | | |

The thickets will be cleared in 21 days.

Capri helps with the first two thickets.

Four thickets

The pattern is now established:

Check to see whether Capri spends all of his time helping clear thicket 1.

If he does, we are finished

If not, even up thickets 1 and 2.

If thicket 3 is finished, Capri helps thickets 1 and 2 equally.

If not, even up thickets 1, 2 and 3.

If thicket 3 is finished, Capri helps thickets 1, 2 and 3 equally.

If not, even up thickets 1, 2 and 3 and then Capri helps all thickets equally.

...

We can now apply this procedure to the remaining data in the question.

B.

| Day | Thicket 1 (47) | | Thicket 2 (37) | | Thicket 3 (27) | | Thicket 4 (17) | |
|-------|----------------|------|----------------|------|----------------|------|----------------|------|
| | Cleared | Rem. | Cleared | Rem. | Cleared | Rem. | Cleared | Rem. |
| 1-10 | 20* | 27 | 10 | 27 | 10 | 17 | 10 | 7 |
| 11-17 | 14* | 13 | 7 | 20 | 7 | 10 | 7 | 0 |
| 18-19 | 4* | 9 | 2 | 18 | 2 | 8 | | |
| 20-27 | 8 | 1 | 16* | 2 | 8 | 0 | | |
| 28 | 1 | 0 | 2* | 0 | | | | |

The thickets will be cleared in 28 days.

Capri helps with the first two thickets.

C.

| Day | Thicket 1 (40) | | Thicket 2 (35) | | Thicket 3 (30) | | Thicket 4 (25) | |
|-------|----------------|------|----------------|------|----------------|------|----------------|------|
| | Cleared | Rem. | Cleared | Rem. | Cleared | Rem. | Cleared | Rem. |
| 1-5 | 10* | 30 | 5 | 30 | 5 | 25 | 5 | 20 |
| 6-10 | 10* | 20 | 5 | 25 | 5 | 20 | 5 | 15 |
| 11-25 | 5 | 15 | 10* | 15 | 5 | 15 | 5 | 10 |
| 16-18 | 6* | 9 | 3 | 12 | 3 | 12 | 3 | 7 |
| 19-21 | 3 | 6 | 6* | 6 | 3 | 9 | 3 | 4 |
| 22-24 | 3 | 3 | 3 | 3 | 6* | 3 | 3 | 1 |
| 25 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 0 |
| 26-27 | 2 | 0 | 2 | 0 | 2 | 0 | | |

The thickets will be cleared in 27 days.

Capri helps with the first three thickets.

Solution 2

We can short-cut the above solutions by doing a little analysis.

If Capri only helped with thicket 1, it would take $t_1/2$ (rounded up) days to clear the thicket.

(We need to round up to allow for t_1 being odd, in which case Capri would not help on the last day.)

If by this time thicket 2 (and therefore all other thickets) has been cleared, Capri will only help with thicket 1. This gives us our first step.

- If $\text{ceiling}(t_1/2) \geq t_2$ then the time to clear all thickets is $\text{ceiling}(t_1/2)$.
(*ceiling* is the rounding-up function.)

If $\text{ceiling}(t_1/2) < t_2$, then Capri should help with thicket 2 as well. This will take $(t_1 + t_2)/3$ days. If this is $\geq t_3$ then Capri will just help with thickets 1 and 2.

It will be useful to introduce a *helpsWith_i* function. This is the time it would take to clear the first *i* thickets with Capri's help.

$$\text{helpsWith}_1 = \text{ceiling}(t_1/2)$$

$$\mathit{helpsWith}_2 = \mathit{ceiling}((t_1 + t_2)/3)$$

$$\mathit{helpsWith}_3 = \mathit{ceiling}((t_1 + t_2 + t_3)/4)$$

...

We can now extend the approach above

| condition | days to clear |
|--------------------------------------|------------------------|
| $\mathit{helpsWith}_1 \geq t_2$ | $\mathit{helpsWith}_1$ |
| else $\mathit{helpsWith}_2 \geq t_3$ | $\mathit{helpsWith}_2$ |
| else $\mathit{helpsWith}_3 \geq t_4$ | $\mathit{helpsWith}_3$ |
| else ... | |

We can now use this approach with the thickets in the question.

- A.** 3 thickets, taking 40, 22 and 16 days.

$$\mathit{helpsWith}_1 = 20, \text{ which is } < t_2,$$

$$\mathit{helpsWith}_2 = 21, \text{ which is } > t_3,$$

Hence $\mathit{helpsWith}_2 = 21$ days.

Capri helps with 2 thickets.

- B.** 4 thickets, taking 47, 37, 27 and 17 days.

$$\mathit{helpsWith}_1 = 24, \text{ which is } < t_2,$$

$$\mathit{helpsWith}_2 = 28, \text{ which is } > t_3,$$

Hence $\mathit{helpsWith}_2 = 28$ days.

Capri helps with 2 thickets.

- C.** 4 thickets, taking 40, 35, 30 and 25 days.

$$\mathit{helpsWith}_1 = 20, \text{ which is } < t_2,$$

$$\mathit{helpsWith}_2 = 25, \text{ which is } < t_3,$$

$$\mathit{helpsWith}_3 = 27, \text{ which is } > t_4,$$

Hence $\mathit{helpsWith}_3 = 27$ days.

Capri helps with 3 thickets.

9. QWERTY

It will be convenient to calculate the number of ways to efficiently type the first letter of each word.

For the bottom row:

Every letter can be reached in 1 jump, so there is only 1 way to do this.

For the middle row: For $\langle \text{spacebar} \rangle \rightarrow S, \dots, K$ there is only one efficient way – a direct jump to the letter. For A and L there is a step before the jump, but in each case just one efficient way $\langle \text{spacebar} \rangle \rightarrow X \rightarrow A$ and $\langle \text{spacebar} \rangle \rightarrow M \rightarrow L$.

For the top row:

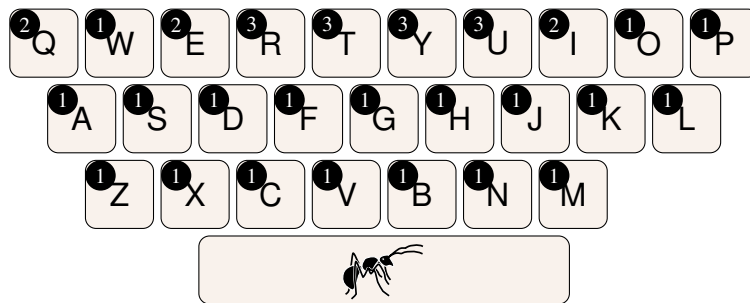
R, T, Y, U can be reached in 3 ways – e.g. $\langle \text{spacebar} \rangle \rightarrow X (C, V) \rightarrow R$

E, I can be reached in 2 ways – e.g. $\langle \text{spacebar} \rangle \rightarrow X (C) \rightarrow E$

W, O can be reached in 1 way – e.g. $\langle \text{spacebar} \rangle \rightarrow X \rightarrow W$

Q can be reached in 2 ways – $\langle \text{spacebar} \rangle \rightarrow X \rightarrow Z (S) \rightarrow Q$

P can be reached in 1 way – $\langle \text{spacebar} \rangle \rightarrow M \rightarrow K \rightarrow P$



We now make the following observations (rules):

1. Antony will jump to the letter to be typed as soon as he is within 2 letters.
Else he would not be efficient.
2. To reach a letter, Antony would not leave a row and then come back to it.
This means that if Antony is on the same row there is only one efficient way to type the next letter.

We use these for the first word.

A. TYPEWRITER

- T 3 from *<spacebar>*
- Y 1 rule 1
- P, E 1 rule 2
- W, R 1 rule 1
- I, T 1 rule 2
- E, R 1 rule 1

There are 3 ways to efficiently type TYPEWRITER

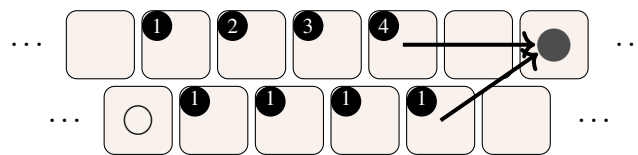
Now consider if Antony is at ○ and his next letter is at ●. He will need 4 steps before jumping. The two possible jumps are shown by arrows.

The numbers in the diagram below show how many ways there are to step to that key from ○.

There is just 1 way to step to the lower jumping key.

There are 4 ways to step to the higher jumping key.

Hence there are $1 + 4 = 5$ ways to efficiently reach ○.



We can generalise this to

- 3 If Antony's next letter is on the next row and is $s - 1$ steps away, there are $s - 1$ ways to reach it.

We use this rule for the next word.

B. ALGORITHM

- A 1 from *<spacebar>*
- L, G 1 rule 2
- O 3 rule 3
- R, I, T 1 rule 2
- H, M 1 rule 1

There are 3 ways to efficiently type ALGORITHM

Now consider if Antony is at ○ and his next letter is at ●. He will need 4 steps before jumping. The three possible jumps are shown by arrows.

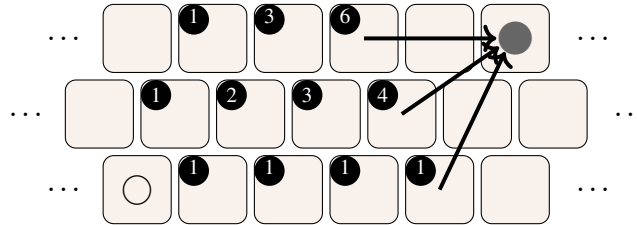
The numbers in the diagram below show how many ways there are to step to that key from ○.

There is just 1 way to step to the lower jumping key.

There are 4 ways to step to the middle jumping key.

There are 6 ways to step to the higher jumping key.

Hence there are $1 + 3 + 6 = 11$ ways to efficiently reach \bigcirc .



We can generalise this to

- 4 If Antony's next letter is on two rows above or below and is 3, 4, 5, .. steps along there are 4, 7, 11, ... ways to reach it.

This can be generalised to

If Antony's next letter is on two rows above or below and is $s \geq 3$ steps along there are $(s - 1)(s - 1)/2 + (s - 1) + 1 = s(s - 1)/2 + 1$ ways to reach it.

- 5 $s = 2$ is a special case. It is not difficult to see that there are 2 ways except for $M \rightarrow P$ for which there is just 1 way.

We use these rules for the remaining word.

C. COMPUTER

| | | |
|---------|----|------------------------------|
| C | 1 | from <i><spacebar></i> |
| O | 11 | rule 4 |
| M | 1 | rule 1 |
| P | 1 | rule 5 |
| U | 1 | rule 2 |
| T, E, R | 1 | rule 1 |

There are 11 ways to efficiently type COMPUTER