

34<sup>th</sup> Annual  
Rowan University  
Programming Contest

hosted by the  
Computer Science Department

Friday, 28<sup>th</sup> April 2023

Contest Problem



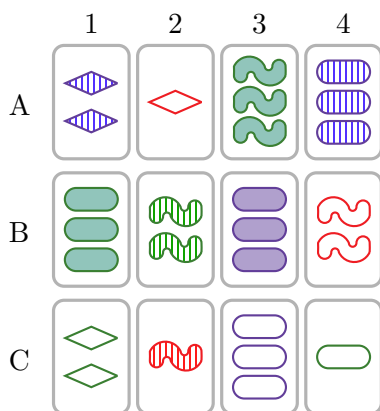
# 1 Introduction

The SET<sup>®</sup> card game\* uses a deck of 81 cards, each with one, two, or three shapes. There are three shapes (diamond, oval, squiggle), which are one of three colors (green, purple, red), in one of three shadings (outlined, striped, solid). Each card thus has four properties: **Number**, **Shading**, **Color**, and **Shape**, and each property has three possible values, as listed in the following chart:

Number	Shading	Color	Shape
One	outlined	green	diamond
Two	striped	purple	oval
Three	solid	red	squiggle

Each property has three possible values. That gives  $3^4 = 81$  possible cards, which is the number of cards in the deck.

The game is played by dealing out 12 cards three rows of four columns, as below (for reference, letters and numbers have been added to the rows and columns, respectively):



A ‘*SET*’ is a group of three cards such that all three cards are either all the same, or all different, with respect to each of the four properties **Number**, **Shading**, **Color**, and **Shape**.

In the deal at left, the cards at {A4, B3, C3} form a *SET*: they match on number, color, and shape, and have different shadings. Also, the cards at {B3, C1, C2} or at {A2, B2, B3}, form a *SET*, because all the cards differ on all four properties.

However, the cards at A1, A2, and A3 are *not* a set, because while they differ on color, number, and shading, The cards at A1 and A2 are the same shape, but both are different from the card at A3.

After the cards are dealt, the players try to identify a *SET* from the twelve cards. The first one to do so says ‘Set’ and removes those cards. Three more cards are dealt out to replace the ones which had been removed, and the players again compete to find another. When all the cards have been dealt, and no more *SETs* can be identified, the player with the most *SETs* wins.

Two complications that must be considered:

1. A card needed for a *SET* may be taken as part of another *SET*.

In the sample deal above, if a player has identified and removed A4, B3, and C3, another player who had seen the *SET* at B3, C1, and C2 is out of luck, because the card at B3 will no longer be there.

2. There may be no *SET* at all in the group of 12 cards.

For any group of 12 cards, there is about a 97% chance that a *SET* can be found. In the event that the players agree after dealing 12 cards that none of them can find a *SET*, three extra cards are dealt out, giving 15. If no *SET* can be found among those 15, three more cards are dealt, giving 18. If still no player can find a *SET*, three more cards are dealt, giving 21. (There is always a *SET* in a grouping of 21 cards.)




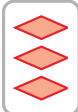
For this year’s programming contest, the challenge is to read in a list of cards to be dealt in a game of SET, and simulate how the game might proceed with those cards.

\*SET is a registered trademark of Cannei, LLC. The distinctive SET symbols and cards are copyrights of Cannei, LLC. All rights reserved. Used with permission from PlayMonster, LLC.

## 2 Notation

### 2.1 Cards

For input and output, we will represent each card with four characters, in this order: number, shading, color, and shape. A list appears below on the left, with examples on the right. (To simplify the input, we are using ‘f’ for ‘Filled’ instead of ‘s’ for ‘Solid’, because we’re using ‘s’ for ‘Striped’.)

Property	Value	Character	Card	Notation	Description
Number	One	1		2spd	2 striped purple diamonds
	Two	2			
	Three	3			
Shading	Filled (Solid)	f		3frs	3 filled (solid) red squiggles
	Outlined	o			
	Striped	s			
Color	Green	g		1ogo	1 outlined green oval
	Purple	p			
	Red	r			
Shape	Diamond	d		3frd	3 filled (solid) red diamonds
	Oval	o			
	Squiggle	s			

(You can see that the characters are just the initials for each value.)

### 2.2 SETs

Because a *SET* does not depend on ordering, any given *SET* can in theory be listed six different ways. For example, these four arrangements are all the same *SET*:



For the purposes of this contest, when a *SET* is listed, it **must** be listed in order as follows:

1. If the cards have different numbers of shapes, they are listed in increasing number.
2. If the cards have the same number of shapes and different shadings, they must be listed in the order Filled (Solid), Outlined, Striped.
3. If the cards have the same number of shapes and same shadings, but different colors, they must be listed in the order Green, Purple, Red.
4. If the cards match on number, shading, and color, they must be listed in the order Diamond, Oval, Squiggle.

Thus, for the *SET* of cards above, the only correct arrangement is the one on the right, and the only correct printed notation for that *SET* is ‘1srs 2ogd 3fpo’.

The correctness of your program’s output depends on getting this ordering right.

### 2.3 Card Placement

For the initial deal of 12 cards, positions will be numbered as in the upper diagram to the right.

1	4	7	10
2	5	8	11
3	6	9	12

In the event no *SETs* are found in the initial grouping of 12, extra cards are added in columns on the right, numbered down vertically.

So, for example, if three more cards were dealt, and still no *SETs* were found, three more would be added giving 18, and the positions would be numbered as in the lower diagram to the right.

1	4	7	10	13	16
2	5	8	11	14	17
3	6	9	12	15	18

Cards should be dealt out in the order specified by the position number. (The first card goes to one, the second to two, and so on.)

When referring to *SETs* which can be identified from a group of cards, they must be ordered based on the lowest sum of their positions. If a *SET* can be made from the cards at 2, 4, and 8, and another from the cards at 1, 5, and 7, the latter should be listed first, because  $2 + 4 + 8 = 14$  while  $1 + 5 + 7 = 13$ .

If two *SETs* have the same sum, such as  $\{1, 5, 8\}$  and  $\{2, 3, 9\}$ , the one with the lowest first card should be listed first. In this case, that would be  $\{1, 5, 8\}$ . If two *SETs* have the same sum and the same lowest card, such as  $\{2, 5, 12\}$  and  $\{2, 6, 11\}$ , then the one with the lowest second card should be listed first.

The correctness of your program's output depends on getting this ordering right.

## 3 Problem Requirements

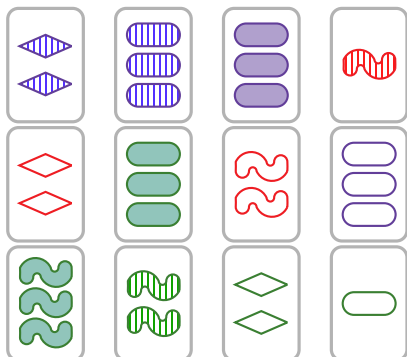
The challenge for this contest is to read in information about the cards in a game of SET, display how the cards will be dealt out, list the *SETs* which can be found, and remove the first one (per the ordering described in Section 2.3). Removed cards should be replaced with remaining cards (if any) until they run out, in increasing order of position.

### 3.1 Example: Displaying Cards Dealt Out

Suppose the input for the program gives only these 12 cards:



Per the placement specification in Section 2.3, they would be dealt out as depicted below on the left, and your program's output would be as depicted below on the right:





Initial Deal:

```

2spd 3spo 3fpo 1srs
2ord 3fgo 2ors 3opo
3fgs 2sgs 2ogd 1ogo
    
```

### 3.2 Example: Displaying *SETs* Found

In the card layout from Section 3.1, there are two *SETs*:

<i>SET</i>	<b>Explanation</b>
	<p>These match on number, color, and shape, and differ on shading.</p>
	<p>These differ on all four properties.</p>

Following Section 2.1, these cards are: {3opo, 3fpo, 3spo} and {2ogd, 3fpo, 1srs}. Per Section 2.2, the first *SET* would be printed as: ‘3fpo 3opo 3spo’, and the second *SET* as ‘1srs 2ogd 3fpo’.

By the specification in Section 2.3, the cards of the first *SET* are in locations 4, 7, and 11, which sum to 22. The cards of the second *SET* are in locations 7, 9, and 10, which sum to 26.

For this grid, your program would print them in the order ‘3fpo 3opo 3spo’ and then ‘1srs 2ogd 3fpo’. (If no *SETs* are found, your program should print ‘No *SETs* found.’)

### 3.3 Example: Removing *SETs*

After your program prints the layout, and any *SETs* which were found, it should remove the first set printed, and then display the configuration again showing which cards were removed. Empty spots should be indicated by the number in Section 2.3, with underscores on each side, as in this example:

<p><b>First set removed:</b></p> <p>2spd __4_ __7_ 1srs          2ord 3fgo 2ors _11_          3fgs 2sgs 2ogd 1ogo</p>	<p>Note that the one digit numbers (4 and 7) have two underscores on the left, but the two-digit number (11) has only one, so that all of the blank spots are filled with four characters.</p>
---	--

### 3.4 Dealing more cards

After a *SET* is removed from a layout of 12 cards (see also Section 3.6), if more cards remain, they should replace the cards that were taken, filling in open spaces in increasing order by number. (For example, in the layout depicted above, the next available card would go into 4, the one after that into 7, and the one after that into 11.)

If, after 12 cards have been dealt out, no *SETs* exist, and there are remaining cards, they should go into positions 13, 14, and 15, as described in Section 2.3. If, with 15 cards, there are still no *SETs*, deal out three more. If, with 18 cards, there are still no *SETs*, deal out three more. (There is always a *SET* in a grouping of 21 cards.) If fewer than three cards remain, deal out all of them.

### 3.5 Clearing cards

It is possible that after a *SET* is removed, no cards remain to be dealt. If the cards that remain include a *SET*, it should be removed and the remaining cards printed. This should continue until no sets remain to be removed.

### 3.6 Compacting the grid

If a layout has more than 12 cards, and a *SET* is removed, the remaining cards should be rearranged to form a grid in the original layout. For example, if there are 15, and you remove 7, 12, and 14, then 8-11 become 7-10, and then 13 becomes 11, and 15 becomes 12. Similarly in any case where there are more than 12 cards.

If the layout is down to 12, and no more cards remain to be dealt, do not compact further. Just leave the empty slots empty.

When all sets have been removed and no cards remain, print a message that the game is over.

## 4 Input

### 4.1 Input Specification

For text input, your program should accept input in the following format:

1. An integer,  $\mathcal{D}$ , where  $1 \leq \mathcal{D} \leq 100$ , which is the number of datasets in this file.
2.  $\mathcal{D}$  data sets, each of which is in this format:
  - (a) One line with one integer,  $\mathcal{C}$ , where  $12 \leq \mathcal{C} \leq 81$ , indicating the number of cards to be dealt.
  - (b)  $\mathcal{C}$  lines, each with four characters, specifying a card.

### 4.2 Sample Input #1

Data in file	Item #	Meaning in plain English
1	1	<i>This file has 1 data set.</i>
12	2.a	<i>Data Set 1 has 12 cards.</i>
2spd	2.b	<i>The 12 cards to be dealt out</i>
2ord		
3fgs		
3spo		
3fgo		
2sgs		
3fpo		
2ors		
2ogd		
1srs		
3opo		
1ogo		

(This input, which corresponds to the sample deal in Section 3.1, is on the website as **sample1.txt**.)

Your program may read the input from the keyboard, or ask the user for a filename and then read the file. GUI-based programs may use text boxes into which the input (or a filename) can be entered, with a buttons to process data. Any reasonable variation in the spirit of the problem is acceptable. You need not do error-checking on the input. Each line will have exactly the number of items described with no stray characters or extra spaces.

All sample and test data sets are available at <http://elvis.rowan.edu/rupc/2023>

## 5 Output

### 5.1 Output Specification

For each data set configuration, your program must generate output as follows:

1. The text ‘Analyzing  $D$  data set(s)’, where  $D$  is the number of data sets in the input.
2. For each data set, print the following:
  - (a) ‘Data Set  $D$ ’, where  $D$  is the number of the data set being reported on.
  - (b) ‘Initial Deal:’, followed by the initial position of all cards, as described in Section 3.1.
  - (c) Either ‘No SETS found.’, if the board contains no *SETs*, or ‘SETs found:  $S$ ’, where  $S$  is the number of sets in the layout, followed by each *SET* found in the cards, as described in Section 3.2.
  - (d) If at least one *SET* was found, print the text ‘First SET removed:’, where the first *SET* is as described the ordering in Section 2.3, and then print the layout again with the cards of that *SET* removed.
  - (e) If more than 12 cards were in the layout when a *SET* was removed, print ‘Layout compacted:’, follow the procedure as described in Section 3.6, and then print the layout again.  
If there were only 12 cards in the layout when a *SET* was removed, do not compact the layout.
  - (f) After removing a *SET* (if any) or failing to find a *SET*, print the words ‘N cards added:’, where  $N$  is the number of cards dealt out, up to three. (If fewer than three cards remain, you can’t add three cards.) If no cards remain, print the words ‘No cards remain.’.  
If you are adding cards to a layout with 12 or more cards, they will be added as described in Section 2.3.  
If you are adding cards to a layout which had 12 when a *SET* was removed, they should be added in the empty spaces left behind by that set, starting with the lowest-numbered space and going up.  
When there are no *SETs* left and no cards remain, print the words ‘The game is over.’.

### 5.2 Sample Output #1 (split across two columns)

Analyzing 1 data set(s) Data Set 1  Initial Deal:  2spd 3spo 3fpo 1srs 2ord 3fgo 2ors 3opo 3fgs 2sgs 2ogd 1ogo  SETs found: 2  3fpo 3opo 3spo 1srs 2ogd 3fpo	First SET removed:  2spd __4_ __7_ 1srs 2ord 3fgo 2ors _11_ 3fgs 2sgs 2ogd 1ogo  No cards remain.  No SETs found.  The game is over.
--	--

This output corresponds to Sample Input #1, on the facing page.

## 6 Sample #2

### 6.1 Sample Input #2

Data in file	Item #	Meaning in plain English
1	1	<i>This file has 1 data set.</i>
27	2.a	<i>Data Set 1 has 27 cards.</i>
1fgo	2.b	<i>The 27 cards to be dealt out</i>
1spo		
1fpo		
1sgo		
2fgo		
2spo		
2fpo		
2sgo		
3fgs		
3frs		
3srd		
3fgd		
3sps		
2fps		
2sgd		
3spd		
3ops		
1sps		
1sgd		
3ogd		
3opo		
2sgs		
1ogs		
1fps		
3fgd		
2sps		
3ors		

(This input is on the website as **sample2.txt**.)

Except for the board, your output **does not** have to duplicate the sample output exactly with respect to spacing or use of upper/lower case. The board should have one blank between cards, and each card printed as four characters with no spacing.

### 6.2 Sample Output #2

Analyzing 1 data set(s)

Data Set 1

Initial Deal:

```
1fgo 1sgo 2fpo 3frs
1spo 2fgo 2sgo 3srd
1fpo 2spo 3fgs 3fgd
```

No SETs found.

3 cards added:

```
1fgo 1sgo 2fpo 3frs 3sps
1spo 2fgo 2sgo 3srd 2fps
1fpo 2spo 3fgs 3fgd 2sgd
```

No SETs found.

3 cards added:

```
1fgo 1sgo 2fpo 3frs 3sps 3spd
1spo 2fgo 2sgo 3srd 2fps 3ops
1fpo 2spo 3fgs 3fgd 2sgd 1sps
```

SETs found: 3

```
1sps 2sgo 3srd
1sps 2spo 3spd
1sps 2fps 3ops
```

First SET removed:

```
1fgo 1sgo 2fpo 3frs 3sps 3spd
1spo 2fgo __8_ _11_ 2fps 3ops
1fpo 2spo 3fgs 3fgd 2sgd _18_
```

Layout compacted:

```
1fgo 1sgo 2fpo 3fgd 2sgd
1spo 2fgo 3fgs 3sps 3spd
1fpo 2spo 3frs 2fps 3ops
```

No SETs found.

*(continued on next page)*



## Sample Output #2, continued (in two columns)

3 cards added:

```
1fgo 1sgo 2fpo 3fgd 2sgd 1sgd
1spo 2fgo 3fgs 3sps 3spd 3ogd
1fpo 2spo 3frs 2fps 3ops 3opo
```

SETs found: 2

```
1spo 2fpo 3opo
1fpo 2spo 3opo
```

First SET removed:

```
1fgo 1sgo __7_ 3fgd 2sgd 1sgd
__2_ 2fgo 3fgs 3sps 3spd 3ogd
1fpo 2spo 3frs 2fps 3ops _18_
```

Layout compacted:

```
1fgo 2fgo 3frs 2fps 3ops
1fpo 2spo 3fgd 2sgd 1sgd
1sgo 3fgs 3sps 3spd 3ogd
```

No SETs found.

3 cards added:

```
1fgo 2fgo 3frs 2fps 3ops 2sgs
1fpo 2spo 3fgd 2sgd 1sgd 1ogs
1sgo 3fgs 3sps 3spd 3ogd 1fps
```

SETs found: 3

```
1fgo 1ogs 1sgd
1fgo 2sgs 3ogd
1ogs 2sgs 3fgs
```

First SET removed:

```
__1_ 2fgo 3frs 2fps 3ops 2sgs
1fpo 2spo 3fgd 2sgd _14_ _17_
1sgo 3fgs 3sps 3spd 3ogd 1fps
```

Layout compacted:

```
1fpo 2spo 3fgd 2sgd 3ogd
1sgo 3fgs 3sps 3spd 2sgs
2fgo 3frs 2fps 3ops 1fps
```

No SETs found.

*(continued on next column)*

3 cards added:

```
1fpo 2spo 3fgd 2sgd 3ogd 3fgd
1sgo 3fgs 3sps 3spd 2sgs 2sps
2fgo 3frs 2fps 3ops 1fps 3ors
```

SETs found: 4

```
1fpo 2sgd 3ors
3fgs 3ors 3sps
1fps 2sps 3ops
1fps 2sgs 3ors
```

First SET removed:

```
__1_ 2spo 3fgd _10_ 3ogd 3fgd
1sgo 3fgs 3sps 3spd 2sgs 2sps
2fgo 3frs 2fps 3ops 1fps _18_
```

Layout compacted:

```
1sgo 3fgs 3sps 3ops 1fps
2fgo 3frs 2fps 3ogd 3fgd
2spo 3fgd 3spd 2sgs 2sps
```

SETs found: 1

```
1fps 2sps 3ops
```

First SET removed:

```
1sgo 3fgs 3sps _10_ _13_
2fgo 3frs 2fps 3ogd 3fgd
2spo 3fgd 3spd 2sgs _15_
```

Layout compacted:

```
1sgo 3fgs 3sps 3ogd
2fgo 3frs 2fps 2sgs
2spo 3fgd 3spd 3fgd
```

No SETs found.

No cards remain.

The game is over.

## 7 Sample #3

### 7.1 Sample Input #3

Data in file	Item #	Meaning in plain English
1	1	<i>This file has 1 data set.</i>
12	2.a	<i>Data Set 1 has 12 cards.</i>
1spd	2.b	<i>The 12 cards to be dealt out</i>
2spd		
3spd		
1fro		
2fro		
3fro		
1ogs		
2ogs		
3ogs		
1sro		
2sro		
3sro		

(This input is on the website as **sample3.txt**.)

### 7.2 Sample Output #3 (split across three columns)

Analyzing 1 data set(s)  
Data Set 1

Initial Deal:

```
1spd 1fro 1ogs 1sro
2spd 2fro 2ogs 2sro
3spd 3fro 3ogs 3sro
```

SETs found: 13

```
1spd 2spd 3spd
1fro 1ogs 1spd
1spd 2fro 3ogs
1spd 2ogs 3fro
1fro 2spd 3ogs
2fro 2ogs 2spd
1ogs 2spd 3fro
1fro 2ogs 3spd
1ogs 2fro 3spd
1fro 2fro 3fro
3fro 3ogs 3spd
1ogs 2ogs 3ogs
1sro 2sro 3sro
```

(continued on column 2)

First SET removed:

```
--1_ 1fro 1ogs 1sro
--2_ 2fro 2ogs 2sro
--3_ 3fro 3ogs 3sro
```

SETs found: 3

```
1fro 2fro 3fro
1ogs 2ogs 3ogs
1sro 2sro 3sro
```

First SET removed:

```
--1_ --4_ 1ogs 1sro
--2_ --5_ 2ogs 2sro
--3_ --6_ 3ogs 3sro
```

SETs found: 2

```
1ogs 2ogs 3ogs
1sro 2sro 3sro
```

(continued on column 3)

First SET removed:

```
--1_ --4_ --7_ 1sro
--2_ --5_ --8_ 2sro
--3_ --6_ --9_ 3sro
```

SETs found: 1

```
1sro 2sro 3sro
```

First SET removed:

```
--1_ --4_ --7_ _10_
--2_ --5_ --8_ _11_
--3_ --6_ --9_ _12_
```

No SETs found.

No cards remain.

The game is over.

## 8 Test Data

The contest website, <<http://elvis.rowan.edu/rupc/2023>>, has four test data files: **test1.txt**, **test2.txt**, **test3.txt**, and **test4.txt**.

Run your program on those files and print the results. **You must submit printed output to earn full points.** Your program will also be run on data known only to the judges.

## 9 Further Reading

Participants in this contest may want to see the book *The Joy of SET: The Many Mathematical Dimensions of a Seemingly Simple Card Game* by Liz McMahon, Gary Gordon, Hannah Gordon, and Rebecca Gordon.