Name(s)_____ Period _____ Date _____

Activity Guide - Storage and Processing

Storage and Processing

Computers use **algorithms** to process information. Algorithms are steps or instructions the computer follows to turn input into output. Computers don't process information exactly like humans, and so their steps might look slightly different from a human's. Even still, the more steps an algorithm takes, the longer it'll take to run.

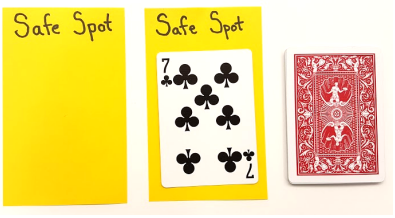
When designing an algorithm you don't just think about the steps of the algorithm. You need to think about the space it will take for a computer to process that information. This activity will help you think about the importance of storage in processing information a little more clearly.

Setup

1. Shuffle a deck of cards
2. Pick up about 10 cards and put them in a stack face down
3. Have a stack of post-its or similarly sized pieces of paper on hand

Rules

1. You can only use one hand through the entire challenge
2. Your hand can hold at most one card
3. You can pull a new card off the deck and look at it whenever you like
4. Once a card leaves your hand it is removed
5. You may not remember anything about cards removed from the game
6. At any time you may make a "safe spot" on the table by placing a post it
7. A safe spot can hold at most one card, face up, preventing it from being removed from play



Challenge 1: Smallest Card

Create an algorithm that always finds the **smallest** card in your pile.

Input: __The Deck__ **Output:** _smallest card__ **Storage:** (How many safe spots did you use?) ___2___

Processing: Write your algorithm in the space below

If both of the safe spots are full, throw away the bigger card.
If there is a safe spot open, put a card from the deck there.
If there are no more cards in the deck, take the smaller card from the safe spot (or the only one if there's only one card). That's your output.

Challenge 2: Largest Card

Create an algorithm that always finds the **largest** card in your pile.

Input: The Deck **Output:** Biggest card **Storage:** (How many safe spots did you use?) 2

Processing: Write your algorithm in the space below

If both of the safe spots are full, throw away the smaller card.
If there is a safe spot open, put a card from the deck there.
If there are no more cards in the deck, take the bigger card from the safe spot (or the only one if there's only one card). That's your output.

Challenge 3: Second Largest Card

Create an algorithm that always finds the **second largest** card in your pile.

Input: The Deck **Output:** 2nd biggest card **Storage:** (How many safe spots did you use?) 3

Processing: Write your algorithm in the space below

If all of the safe spots are full, throw away the smaller card.
If there is a safe spot open, put a card from the deck there.
If there are no more cards in the deck, take the middle card from the safe spot (or the smaller one if there are only two cards). That's your output.

Challenge 4: Middle Card

Create an algorithm that always finds the **middle** or median card in your pile (the one that would be in the middle if you lined up all your cards in number order). You can assume you have an odd number of cards.

Input: The Deck **Output:** middle card **Storage:** (How many safe spots did you use?) 10

Processing: Write your algorithm in the space below

Put all your cards in the safe spots.
Take the biggest card, then the smallest card, until you only have one card left. The card that is left is your output.