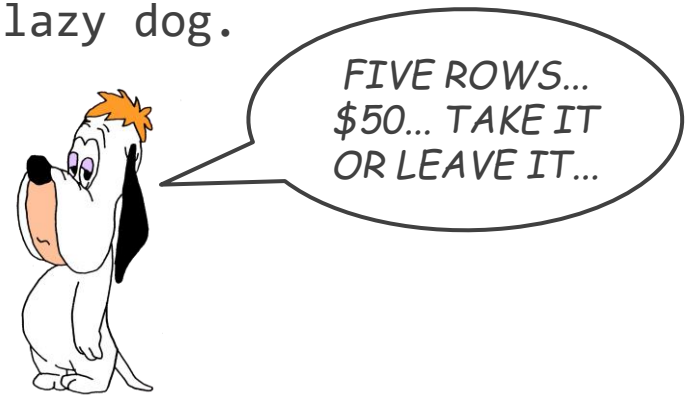
Copying Conundrum

- Nick is trying to design an advertisement, but Droopy is charging him for every row of text he uses! Each row can fit 10 characters, including letters, spaces, and punctuation. Nick needs to place the text in order, without splitting any words across rows. His goal is to use the fewest number of rows possible to keep the costs low and avoid paying too much to Droopy.
- Suppose the text Nick needs to fit into his advertisement is:

```
The quick brown fox jumps over the lazy dog.
```

- And here is what he did:

T	h	e		q	u	i	c	k	
b	r	o	w	n		f	o	x	
j	u	m	p	s					
o	v	e	r		t	h	e		
l	a	z	y		d	o	g	.	



Think-pair-share: Describe a greedy strategy that Nick can use to minimize the cost

Copying Conundrum

- Nick is trying to design an advertisement, but Droopy is charging him for every row of text he uses! Each row can fit 10 characters, including letters, spaces, and punctuation. Nick needs to place the text in order, without splitting any words across rows. His goal is to use the fewest number of rows possible to keep the costs low and avoid paying too much to Droopy.
- Suppose the text Nick needs to fit into his advertisement is:

```
The quick brown fox jumps over the lazy dog.
```

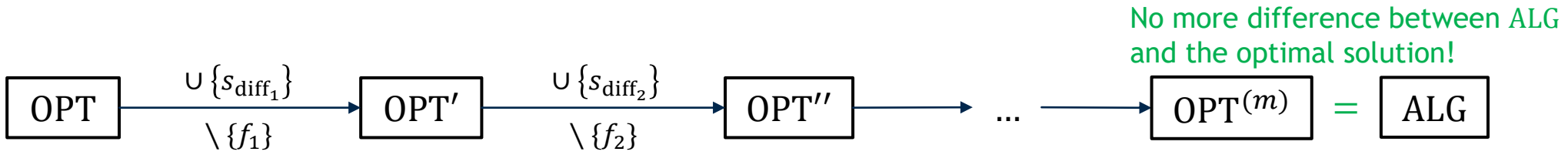
- And here is what he did:

T	h	e		q	u	i	c	k	
b	r	o	w	n		f	o	x	
j	u	m	p	s					
o	v	e	r		t	h	e		
l	a	z	y		d	o	g	.	

- **Greedy strategy:** Fill each row with as many words as possible without exceeding the 10-character limit. Move to the next row and repeat until the sentence is complete.

Induction and Exchange Argument

- **Goal:** Prove that ALG *is* an optimal solution
- **Exchange argument:** Show that we can transform any optimal solution into the solution given by our algorithm by exchanging each piece of it out one-by-one without increasing the final cost



- Induction framing for proving optimality
  - **Base case:** simplest form of the problem – often zero
  - **Inductive Hypothesis:** Assume that the first $k-1$ choices of the greedy solution are part of *some* optimal solution
  - **Inductive step:** Show that the first k choices (collectively, not just the k^{th}!) are also part of *some* optimal solution (could be different from the one in IH)

Copying Conundrum: Key Observations

- **Greedy strategy:** Fill each row with as many words as possible without exceeding the 10-character limit. Move to the next row and repeat until the sentence is complete.

T	h	e		q	u	i	c	k	
b	r	o	w	n		f	o	x	
j	u	m	p	s		o	v	e	r
t	h	e		l	a	z	y		
d	o	g	.						

Optimal Solution A (Greedy)

T	h	e		q	u	i	c	k	
b	r	o	w	n		f	o	x	
j	u	m	p	s					
o	v	e	r		t	h	e		
l	a	z	y		d	o	g	.	

Optimal Solution B (Nick's)

T	h	e		q	u	i	c	k	
b	r	o	w	n		f	o	x	
j	u	m	p	s		o	v	e	r
t	h	e							
l	a	z	y		d	o	g	.	

Optimal solution C

- Observing **first** difference:
  - A vs B: 'over' in A is placed one row earlier than in B
  - A vs C: 'lazy' in A is placed one row earlier than in C

Copying Conundrum Exchange Step

- Notation: Here, r_i means placing word i on row r_i

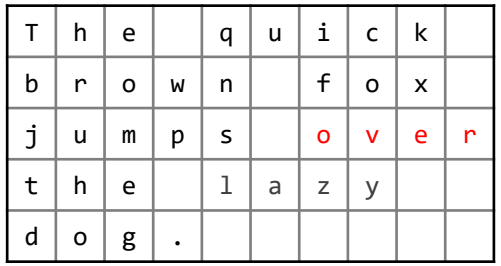
  $\text{ALG} =$ $r_1, \ldots, r_n$ = assignment by GREEDYASSIGN ; $\text{OPT} =$ $o_1, \ldots, o_n$ = some optimal assignment

- If $\text{ALG} = \text{OPT}$, done. Otherwise,
  - Let diff be the first index where $r_{\text{diff}} \neq o_{\text{diff}}$ (this means $r_1 = o_1, \ldots, r_{\text{diff}-1} = o_{\text{diff}-1}$)
  - Modify OPT to agree with ALG up to r_{diff} (this involves changing some o_j(s) that ~~is~~ not in $r_1, \ldots, r_{\text{diff}-1}$)
  - Let's call this modified solution OPT' TODO 1: Identity o_j(s)/ Describe OPT' — is/are
- Now, we need to prove that
  - OPT' is still a **valid** solution TODO 2: Show that OPT' doesn't violate any rules
    - In order
    - Can't split words
  - OPT' is still an **optimal** solution TODO 3: Show that $o'_n = o_n$

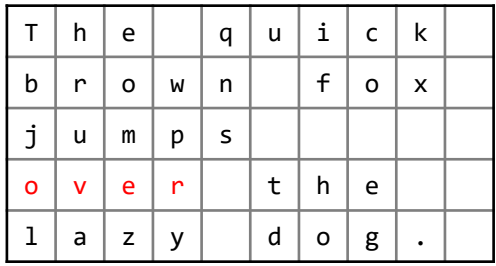
Your turn: Work on the three TODOs in group

Copying Conundrum Exchange Step

- First, observe that $r_{\text{diff}} < o_{\text{diff}}$ by the design of GREEDYASSIGN
- Specifically, $r_{\text{diff}} = o_{\text{diff}} - 1$ (from the observation earlier!)
- Construct $\text{OPT}' = r_1, .., r_{\text{diff}}, o_{\text{diff+1}}, \dots, o_n$
- **Validity:**
  - By the design of GreedyAssign, placing word diff on row r_{diff} will not overflow the row
  - Did not split any words in OPT'
- **Optimality:**
  - OPT' uses o_n rows, which is assumed to be optimal (by definition of OPT)

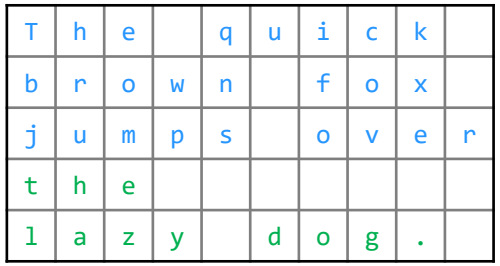


ALG



OPT

$$r_{\text{diff}} = 3 < 4 = o_{\text{diff}}$$



OPT'