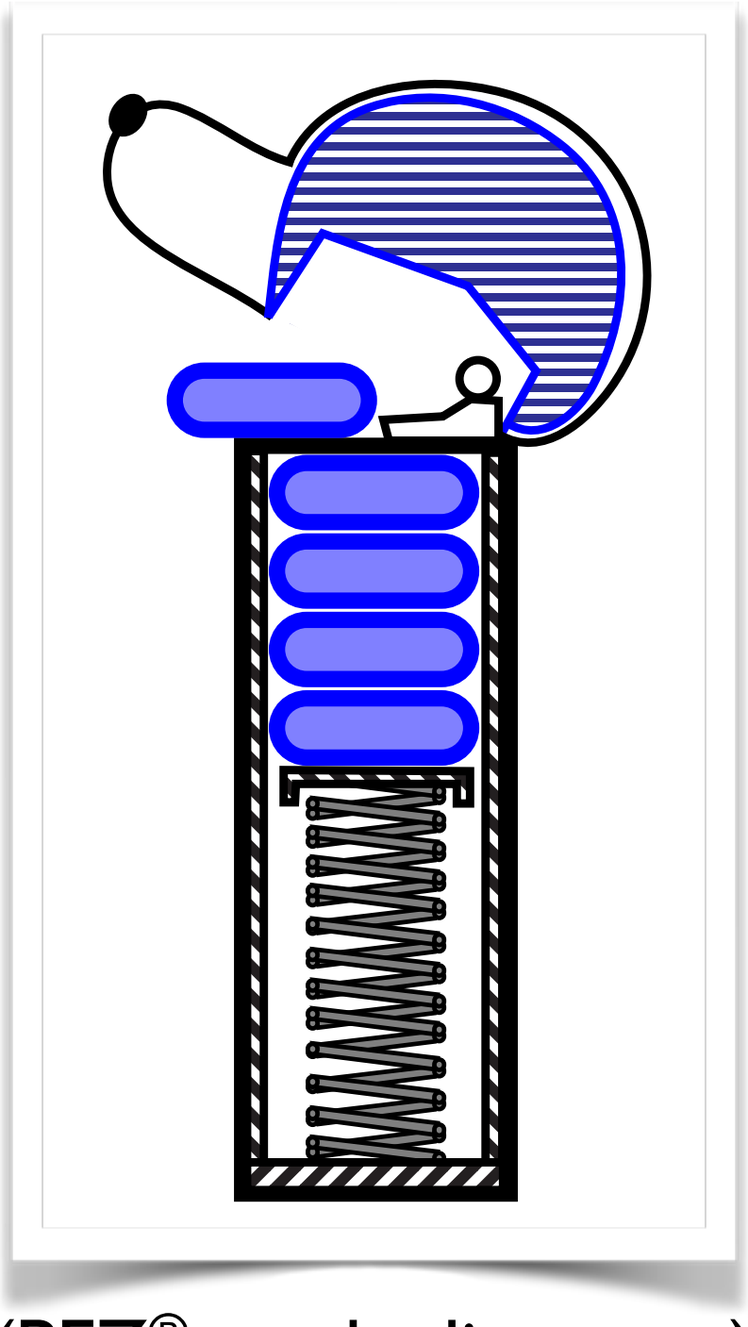
Winter 2016
COMP-250: Introduction to Computer Science

Lecture 7, February 2, 2016

Stack ADT



(PEZ® candy dispenser)

ADTs

- An Abstract Data Type is an abstraction of a data structure: no coding is involved.
- The ADT specifies:
  - what can be stored in it
  - what operations can be done on/by it.
- There are lots of formalized and standardized ADTs (in Java).

ADTs

- For example, if we are going to model a bag of marbles as an ADT, we could specify that
  - this ADT stores marbles
  - this ADT supports putting in a marble and getting out a marble.
- In this course we are going to learn a lot of different standard ADTs. (stacks, queues, trees...)
- (A bag of marbles is not one of them.)

Stack

- A stack is a container of objects that are inserted and removed according to the last-in-first-out (LIFO) principle.
- Objects can be inserted at any time, but only the last (the most-recently inserted) object can be removed.
- Inserting an item is known as "pushing" onto the stack.
- "Popping" off the stack is synonymous with removing an item.

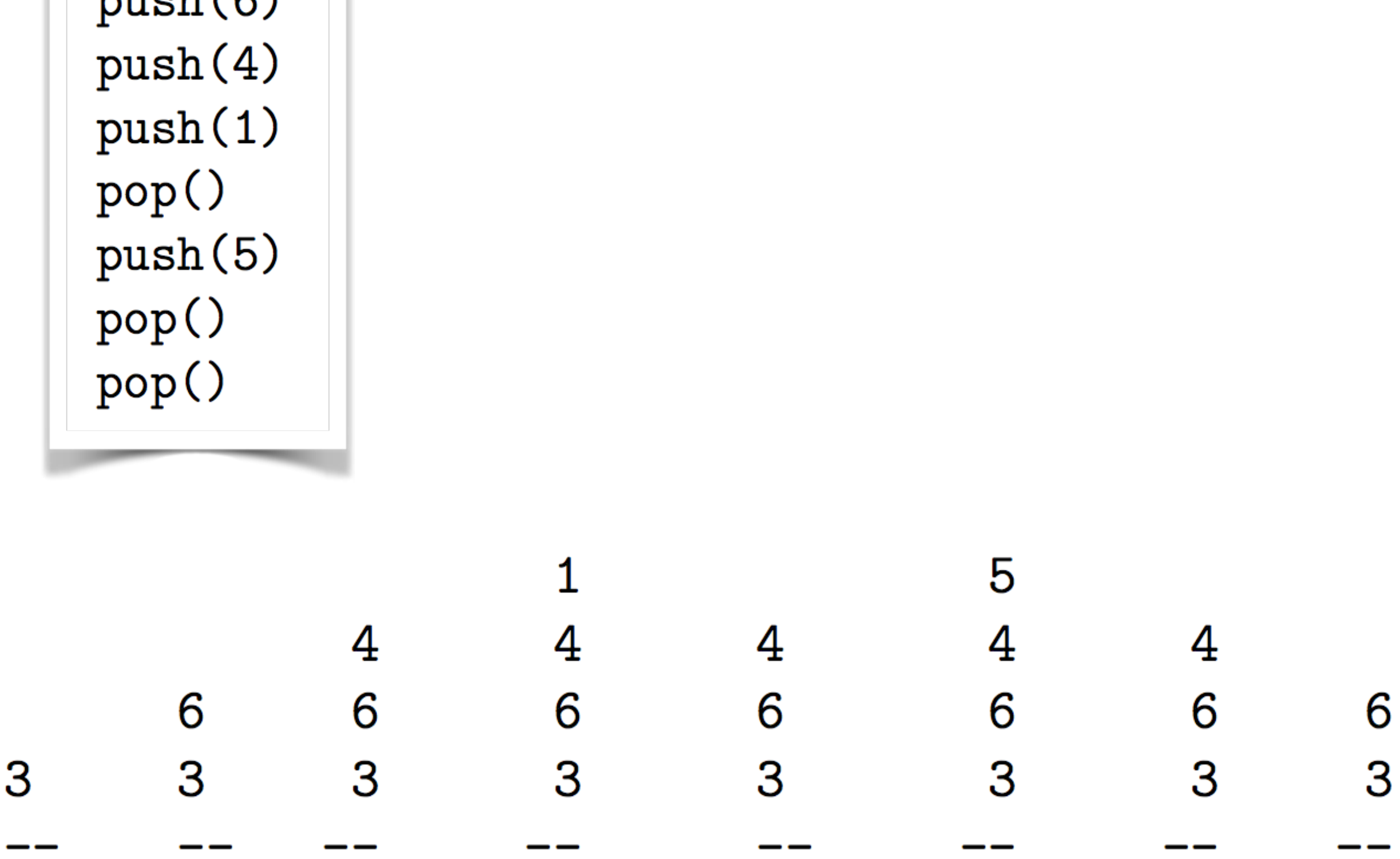
Stack

- A stack is an ADT that supports two main methods:
  - push(*o*): Inserts object *o* onto top of stack
  - pop(): Removes the top object of stack and returns it; if the stack is empty then an error occurs.

- The following support methods should also be defined:
  - size(): returns the number of objects in stack
  - isEmpty(): returns a boolean indicating if stack is empty.
  - top(): returns the top object of the stack, without removing it; if the stack is empty then an error occurs.

```
push(3)
push(6)
push(4)
push(1)
pop()
push(5)
pop()
pop()
```

Examples:

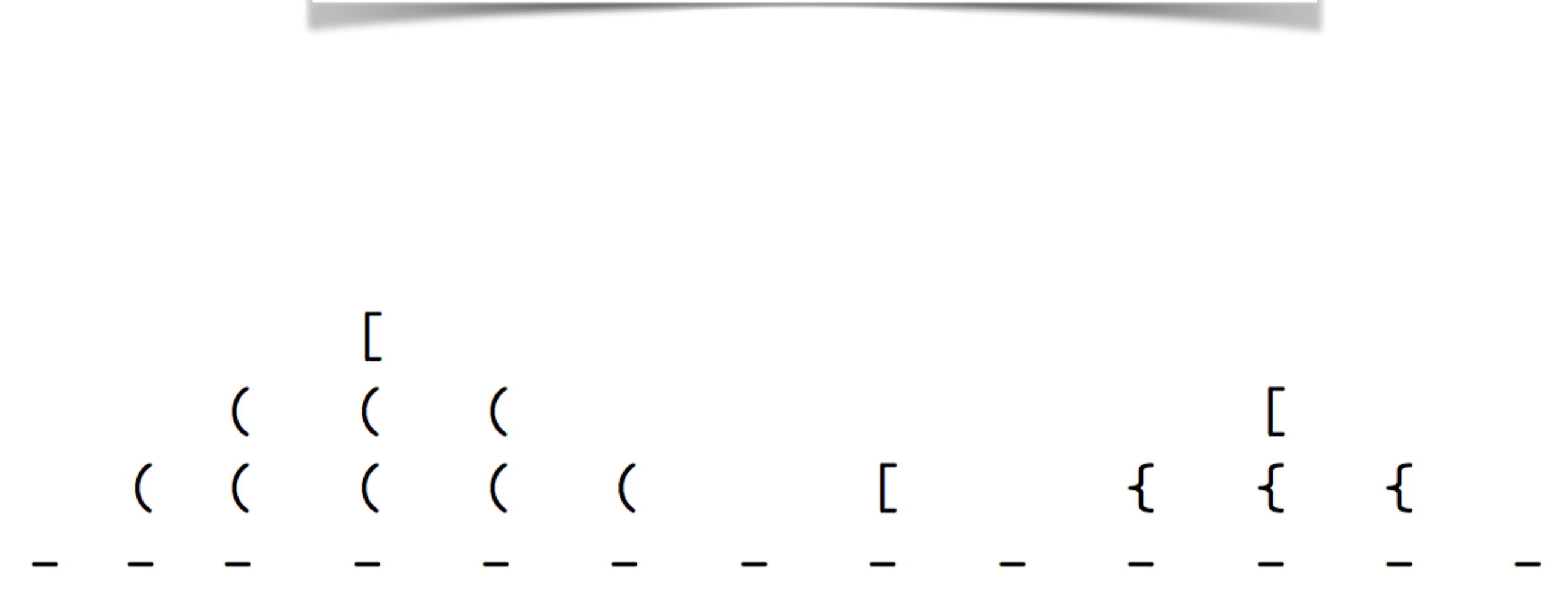


Examples:

```
3 + (4 - x)* 7 + (y - 2 * (2 + x)).
```

Examples:

(([ ])) [ ] { [ ] }



Examples:

(([ )]) [[ ]] { [}]

```
          [
        (   (
      (   (   (
--- --- --- ---  X  since next symbol is ")" which doesn't match top
```

Examples:

Computer Science COMP 250
Introduction to Computer Science

Winter 2016, Schedule: TTh 13:05-14:25; STBIO S1/4

HW-1 due-date postponed to Feb 4th.

Description: (3 credits; 3 hours) An introduction to the design of computer algorithms, including basic data structures, analysis of algorithms, and establishing correctness of programs. Overview of topics in computer science.

Prerequisite: Familiarity with a high level programming language and CEGEP level mathematics. Computer Science Major and Honours students are advised to take MATH 240 simultaneously with COMP 250 or with COMP 251. Although it not a prerequisite either, COMP 202 will provide you a solid background for programming in Java.

Instructor: Prof. Claude Crépeau
Office Hours: Friday 12:00-15:00, McConnell 110N, (514) 398-4716
avoid sending me HomeWork questions directly; uses links below for faster replies...

FOR ALL CLASS MATTERS CONTACT US
via e-mail: *cs250@cs.mcgill.ca*
or *Discussion board: cs250qanda.cs.mcgill.ca/* NOW WORKING; you will need a cs.mcgill.ca account to use it.

Examples:

Computer Science COMP 250
Introduction to Computer Science

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
<TITLE>McGill University School of Computer Science: COMP 250 (sec 1)
</TITLE>
</HEAD>

<BODY TEXT="#333333" BGCOLOR="#FFFFFF" BACKGROUND="http://www.cs.mcgill.ca/~crepeau/GIF/mcgill-emboss-blur.gif">

<center>
<H1>Computer Science COMP 250<BR>
Introduction to Computer Science</H1>
<FONT SIZE=+2>
<P>Winter 2016, Schedule: TTh 13:05-14:25; STBIO S1/4 </P>
<BR>
<FONT color=red>HW-1 due-date postponed to Feb 4th.</FONT>
</FONT>
</center>
<HR>

<P><B>Description</B>: (3 credits; 3 hours)
An introduction to the design of computer algorithms, including basic data structures,
analysis of algorithms, and establishing correctness of programs.
Overview of topics in computer science.
<BR>
<BR>
<B>Prerequisite</B>: Familiarity with a high level programming language and CEGEP level mathematics. Computer Science Major and Honours students are advised to take MATH 240
simultaneously with COMP 250 or with COMP 251. Although it not a prerequisite either, COMP 202 will provide you a solid background for programming in Java.
<HR>

<UL>
<P>
<B>Instructor:</B>
<a href=http://www.cs.mcgill.ca/~crepeau>
Prof. Claude Cr&eacute;peau</a><BR>
Office Hours: Friday 12:00-15:00, McConnell 110N, (514) 398-4716
<BR>
<FONT color=red> avoid sending me HomeWork questions directly; uses links below for faster replies...</FONT>
<BR>
<BR>
FOR ALL CLASS MATTERS CONTACT US
<BR> via e-mail:
<I><A HREF="mailto:cs250@cs.mcgill.ca">cs250@cs.mcgill.ca</A></I>
<BR>
or
<I> Discussion board: </I>
<I><A HREF=https://cs250qanda.cs.mcgill.ca/>cs250qanda.cs.mcgill.ca/</A></I>
NOW WORKING; you will need a cs.mcgill.ca account to use it.<br>
```

Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT > </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 …

<P> <FONT>
<H1> <FONT><FONT><FONT><FONT><FONT>
<TITLE>
<HEAD><HEAD><HEAD> <BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY>
<HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML>

_____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____ _____

Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .

<P> <FONT>
<H1> <FONT><FONT><FONT><FONT><FONT>
<TITLE>
<HEAD><HEAD><HEAD> <BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY>
<HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML>

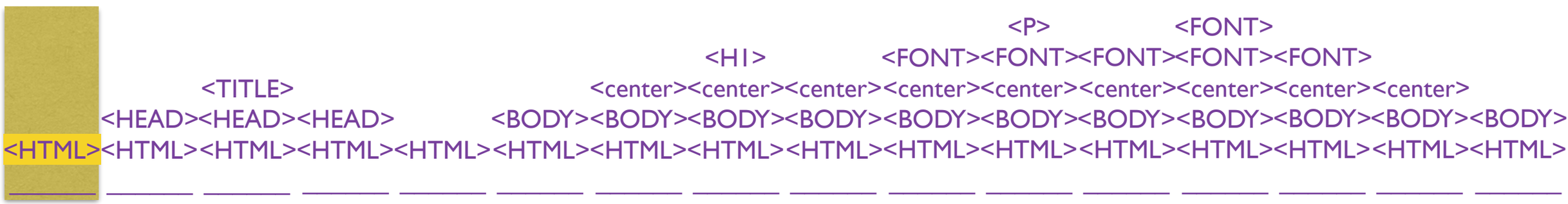
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



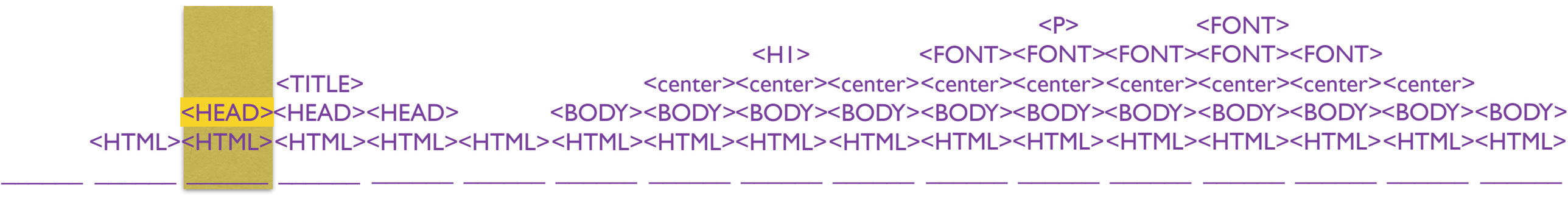
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



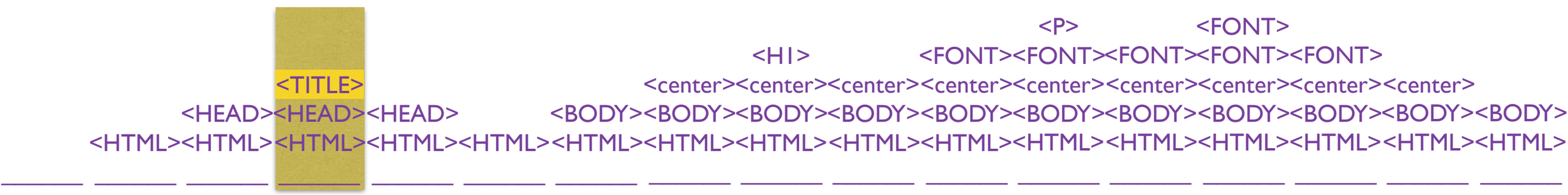
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



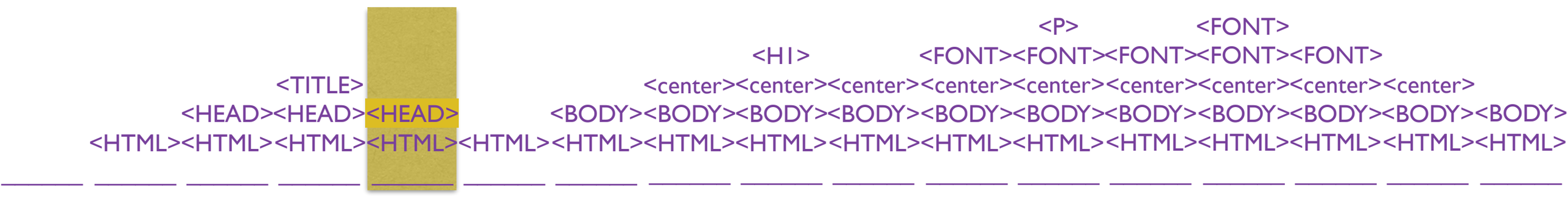
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



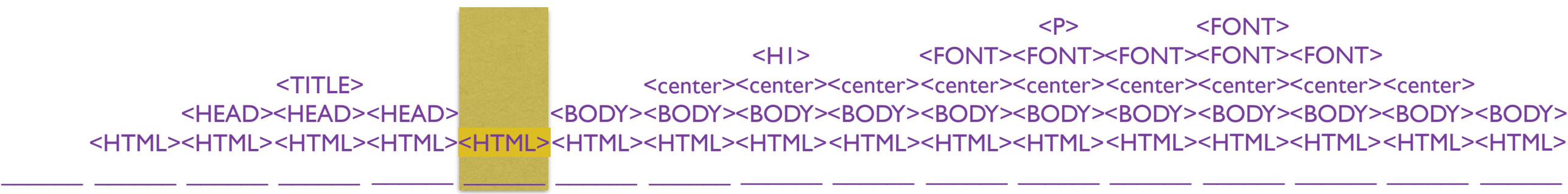
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



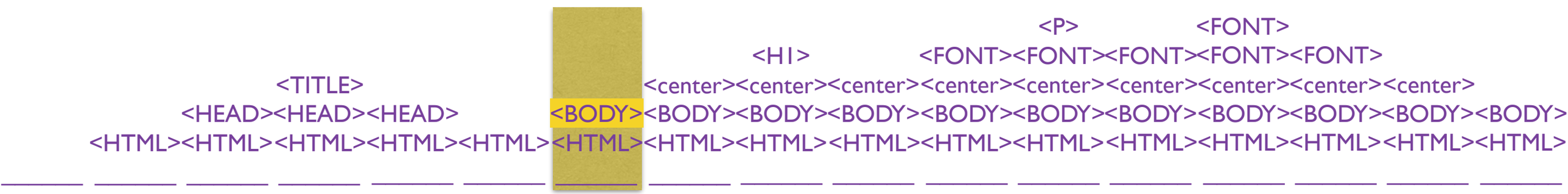
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



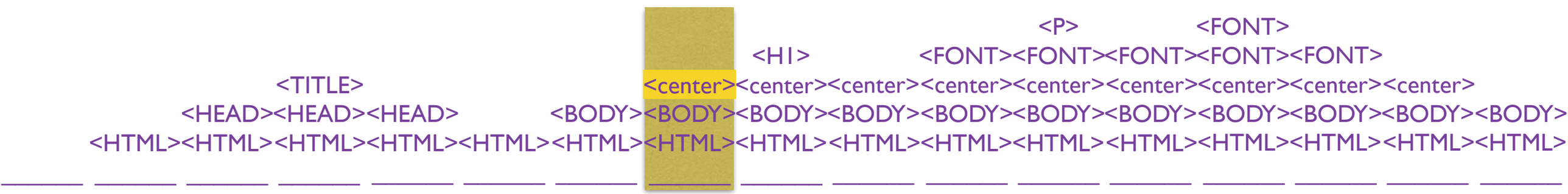
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



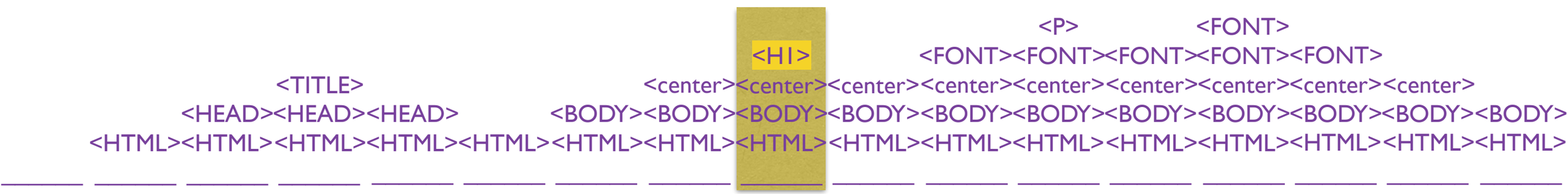
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



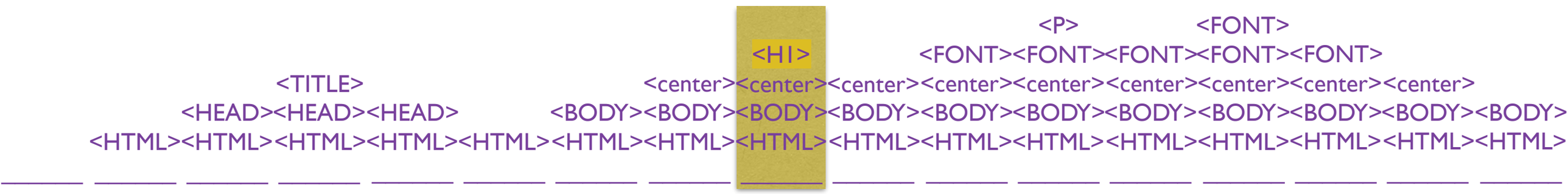
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



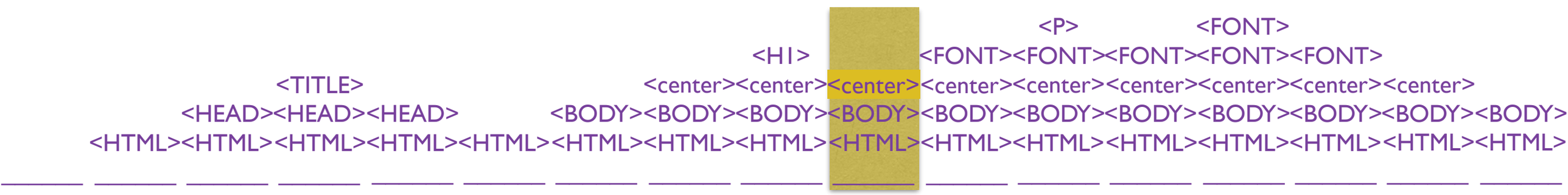
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



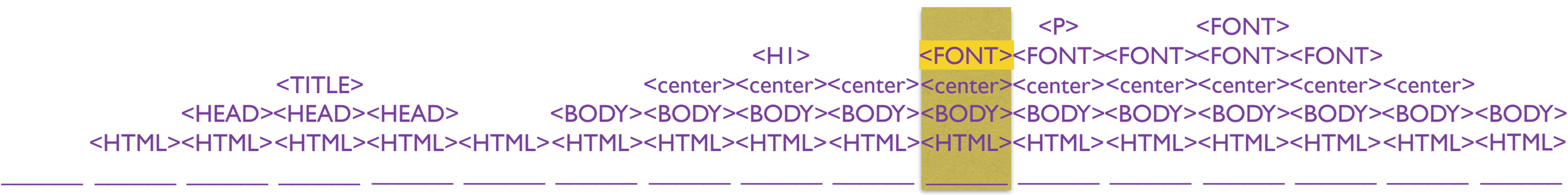
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .

<P>
<FONT>
<H1> <FONT><FONT><FONT><FONT><FONT>
<TITLE>
<HEAD><HEAD><HEAD> <BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY>
<HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML>

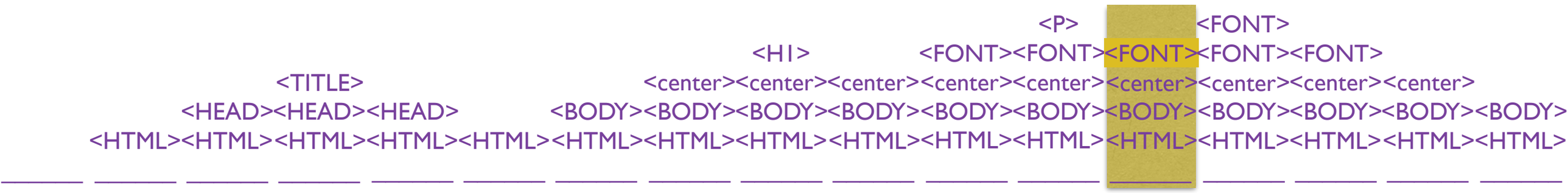
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



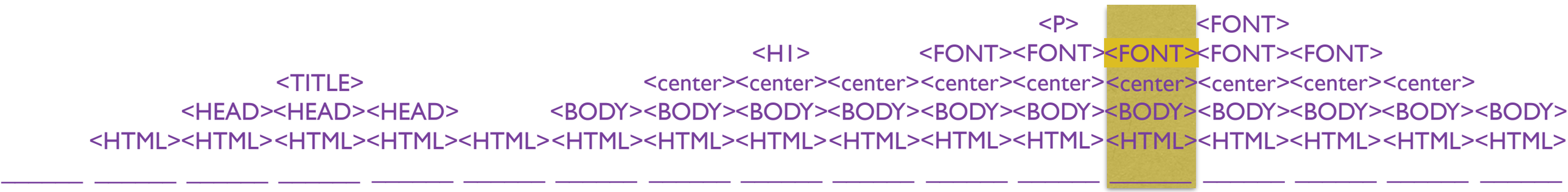
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



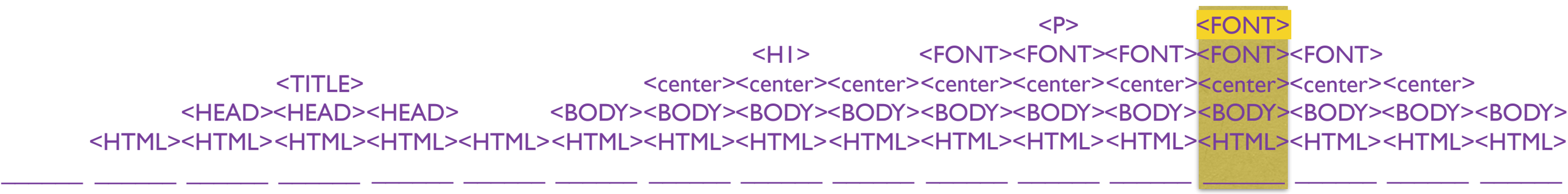
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .

<P> <FONT>
<H1> <FONT><FONT><FONT><FONT><FONT>
<TITLE>
<HEAD><HEAD><HEAD> <BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY>
<HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML>

Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT>
 <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P>
<B> </B> <a>

 <FONT> </FONT>

 <I>
<A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .

<P> <FONT>
<H1> <FONT><FONT><FONT><FONT><FONT>
<TITLE>
<HEAD><HEAD><HEAD> <BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY><BODY>
<HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML><HTML>

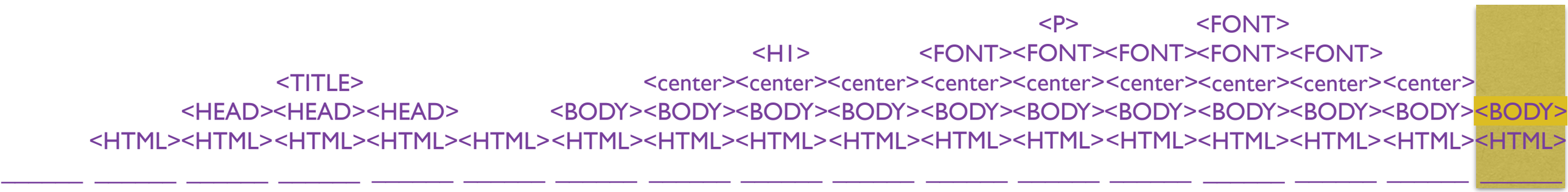
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 …



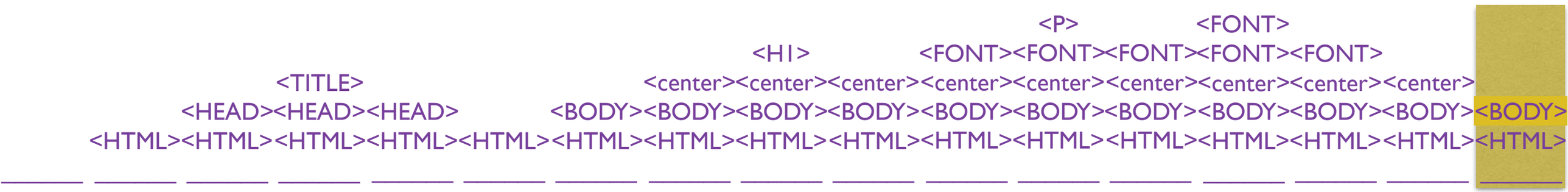
Examples:

<!> <HTML> <HEAD> <TITLE> </TITLE> </HEAD> <BODY> <H1>
 </H1> <FONT> <P> </P>
 <FONT> </FONT> </FONT> <HR> <P> <B> </B>

 <B> </B> <HR> <UL> <P> <B> </B> <a>

 <FONT> </FONT>

 <I> <A> </A> </I>
 <I> </I> <I> <A> </A> </I>
 . . .



Examples:

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

Examples:

```
                  -   -   -
          (   (   (   (   (       *   *
  +   +   +   +   +   +   +   +   +   +       +
__ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __
```

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

```
                      1   1               7
              4   4   4   4   3   3   3  21
  3   3   3   3   3   3   3   3   3   3   3 24 24
__ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __
```

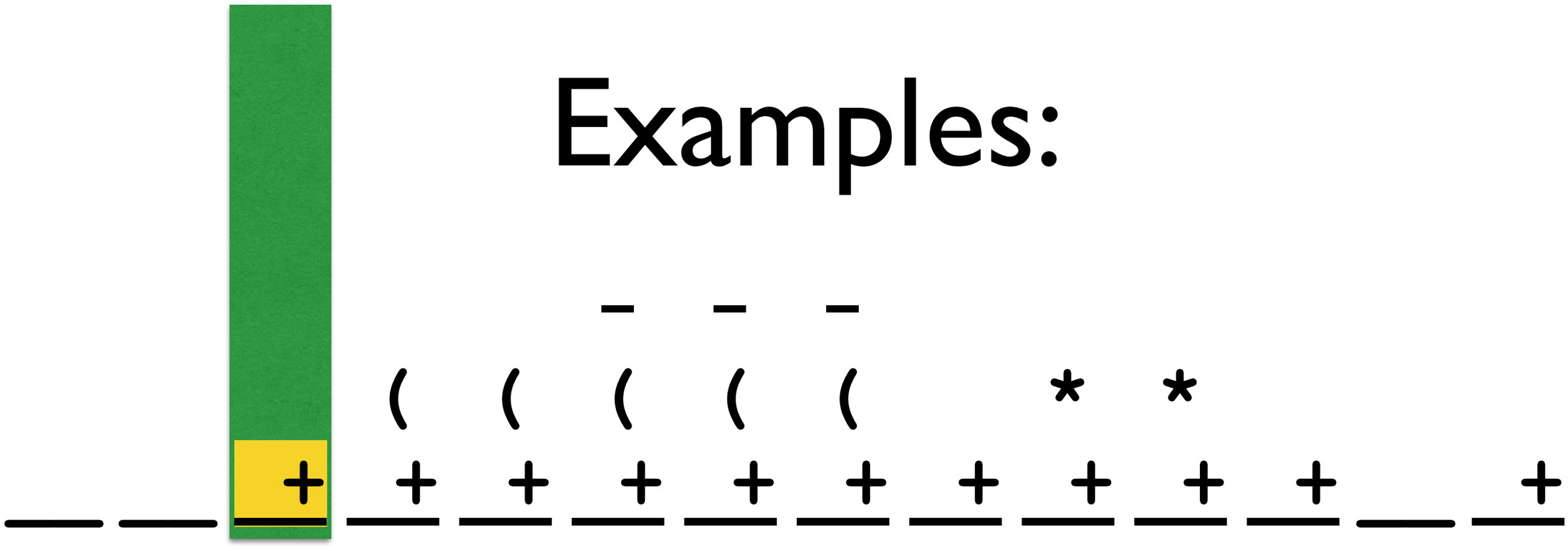
Examples:

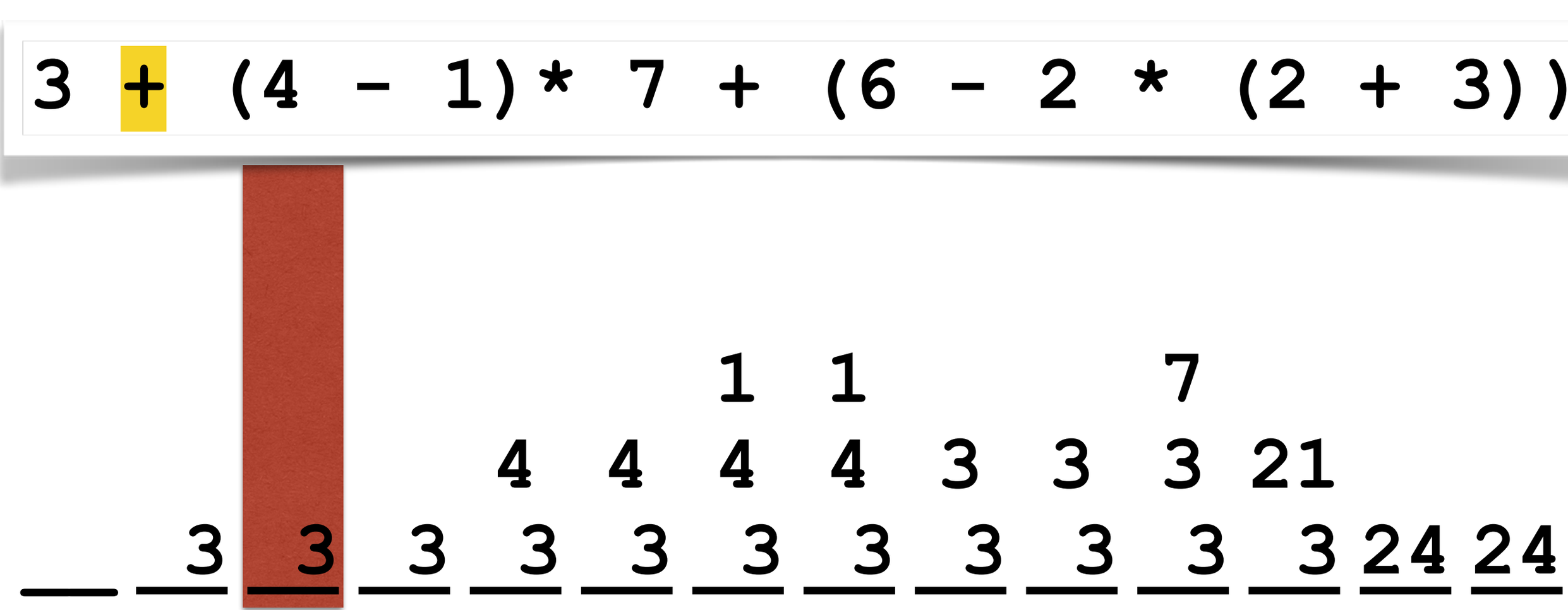
					-	-	-						
			(	(	(	(	(		*	*			
	+	+	+	+	+	+	+	+	+	+			+

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

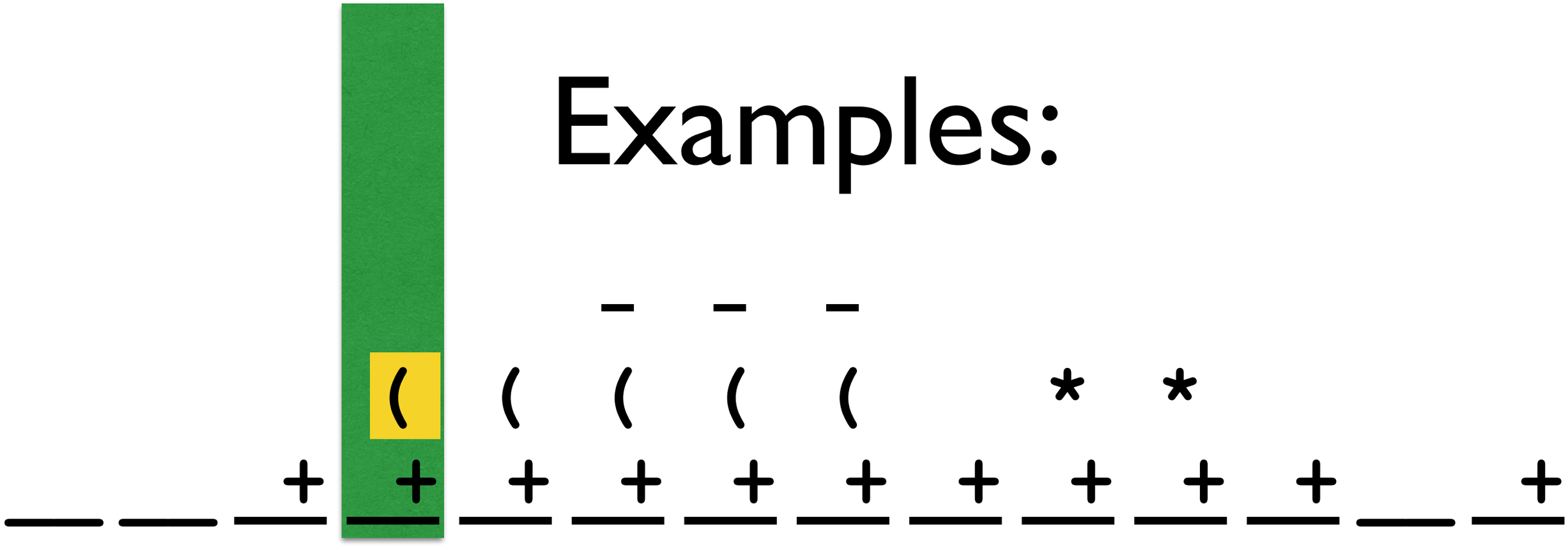
					1	1			7				
			4	4	4	4	3	3	3	21			
	3	3	3	3	3	3	3	3	3	3	24	24	

Examples:

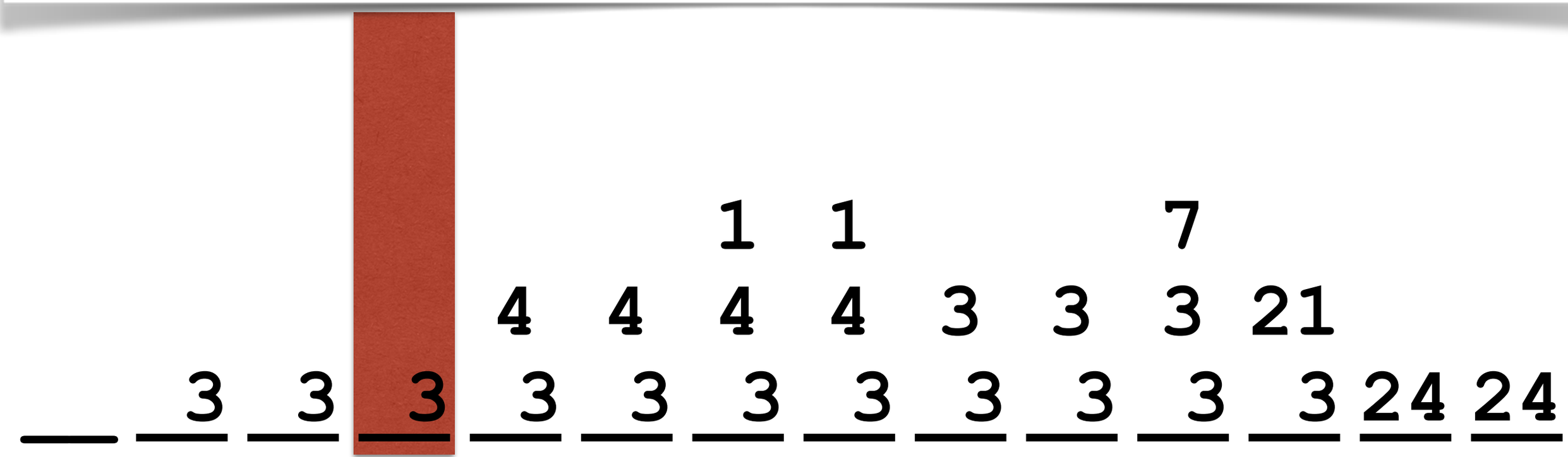




Examples:




```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```



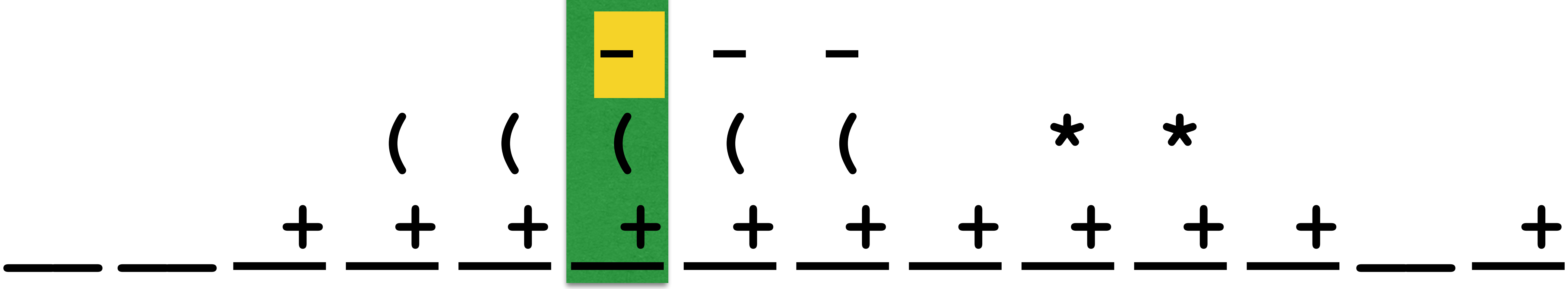
Examples:

```
                  -   -   -
          (   (   (   (   (       *   *
      +   +   +   +   +   +   +   +   +   +       +
__ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __
```

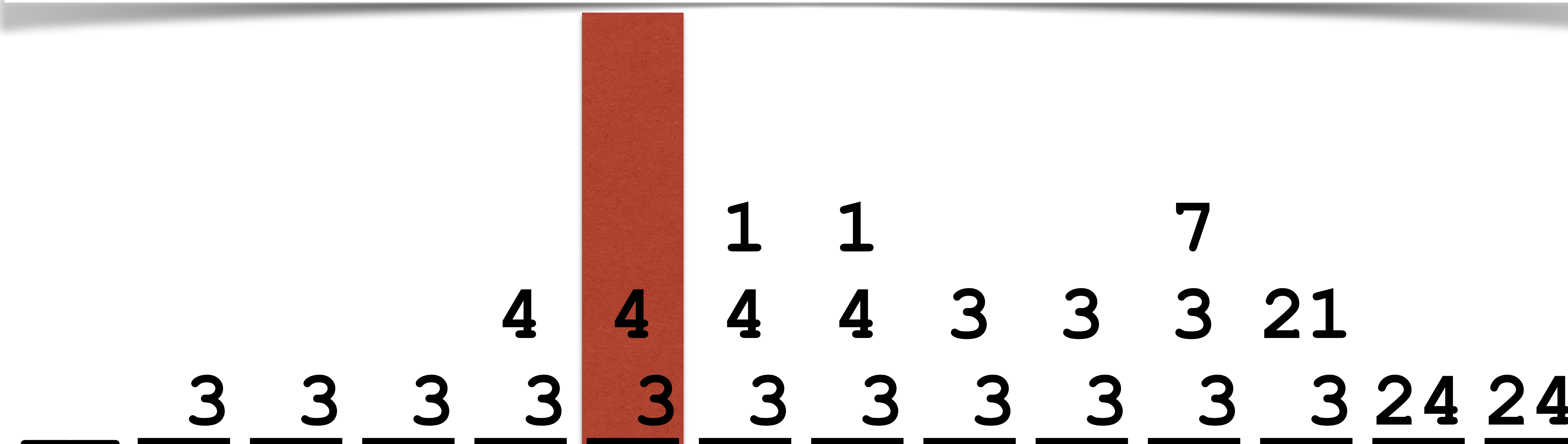
```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

```
                      1   1               7
          4   4   4   4   3   3   3   21
      3   3   3   3   3   3   3   3   3   3   24  24
__ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __ __
```

Examples:




```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```



Examples:

					-	-	-							
			(	(	(	(	(		*	*				
		+	+	+	+	+	+	+	+	+	+			+
_	_	_	_	_	_	_	_	_	_	_	_	_	_	_

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

						1	1			7				
				4	4	4	4	3	3	3	21			
	3	3	3	3	3	3	3	3	3	3	3	24	24	
_	_	_	_	_	_	_	_	_	_	_	_	_	_	_

Examples:

					-	-	-						
			(	(	(	(	(			*	*		
		+	+	+	+	+	+	+	+	+	+		+
_	_	_	_	_	_	_	_	_	_	_	_	_	_

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

					1	1				7			
				4	4	4	4	3	3	3	21		
	3	3	3	3	3	3	3	3	3	3	3	24	24
_	_	_	_	_	_	_	_	_	_	_	_	_	_

Examples:

						-	-	-					
			(	(	(	(	(		*	*			
		+	+	+	+	+	+	+	+	+	+		+
_	_	_	_	_	_	_	_	_	_	_	_	_	_

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

4-1=3

						1	1			7			
				4	4	4	4	3	3	3	21		
	3	3	3	3	3	3	3	3	3	3	3	24	24
_	_	_	_	_	_	_	_	_	_	_	_	_	_

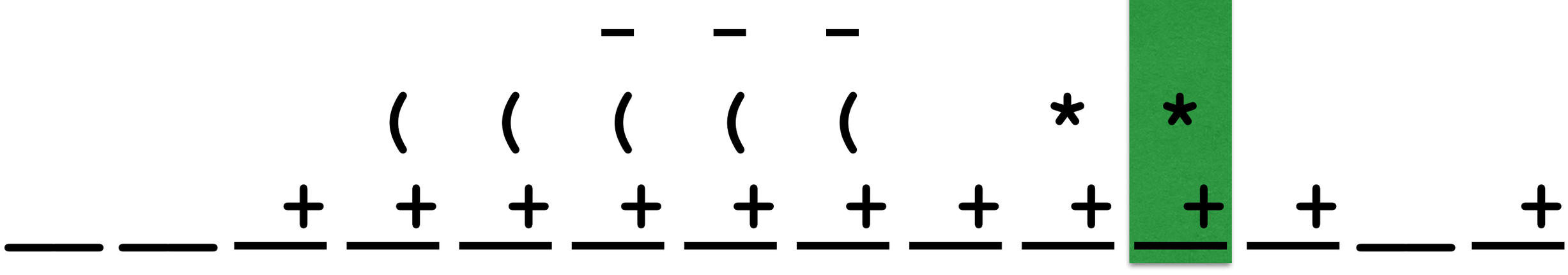
Examples:

						-	-	-						
				(	(	(	(	(		*	*			
		+	+	+	+	+	+	+	+	+	+		+	

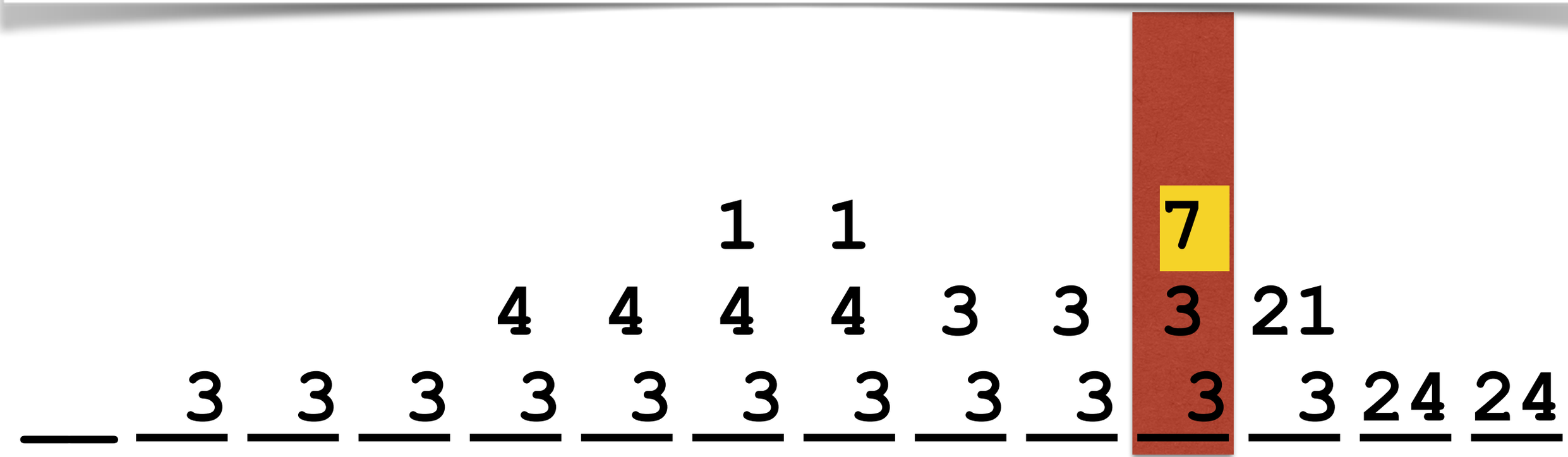
```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

						1	1			7				
				4	4	4	4	3	3	3	21			
	3	3	3	3	3	3	3	3	3	3	3	24	24	

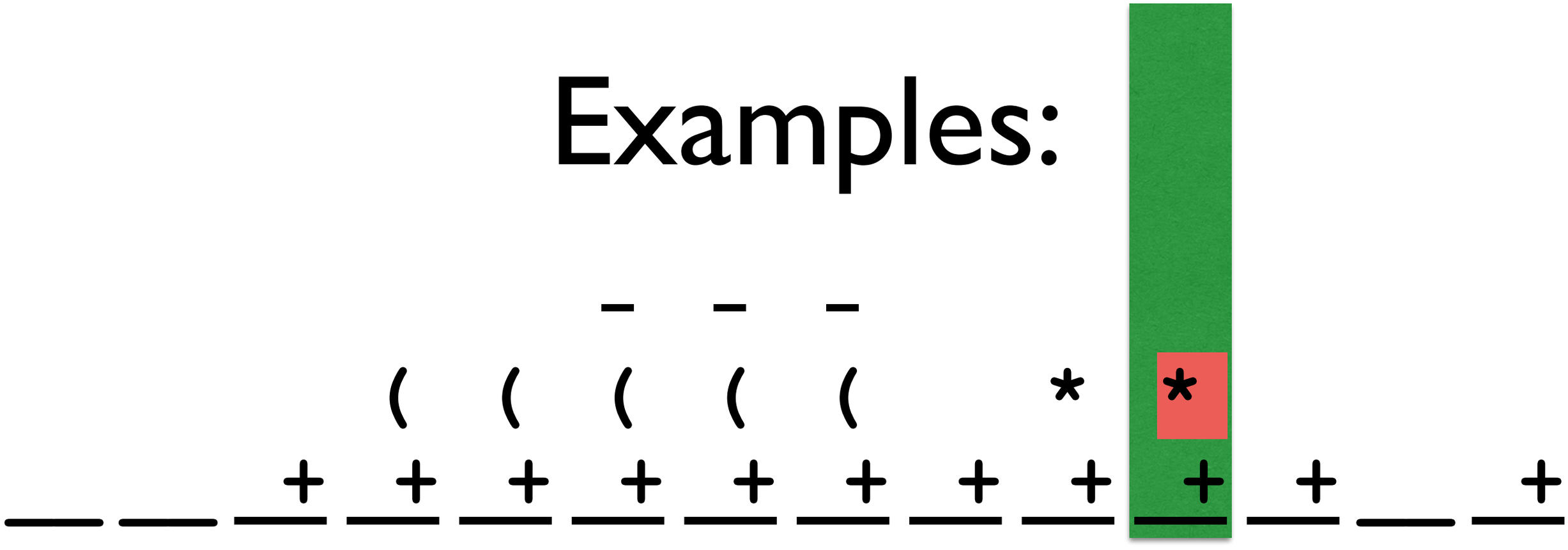
Examples:



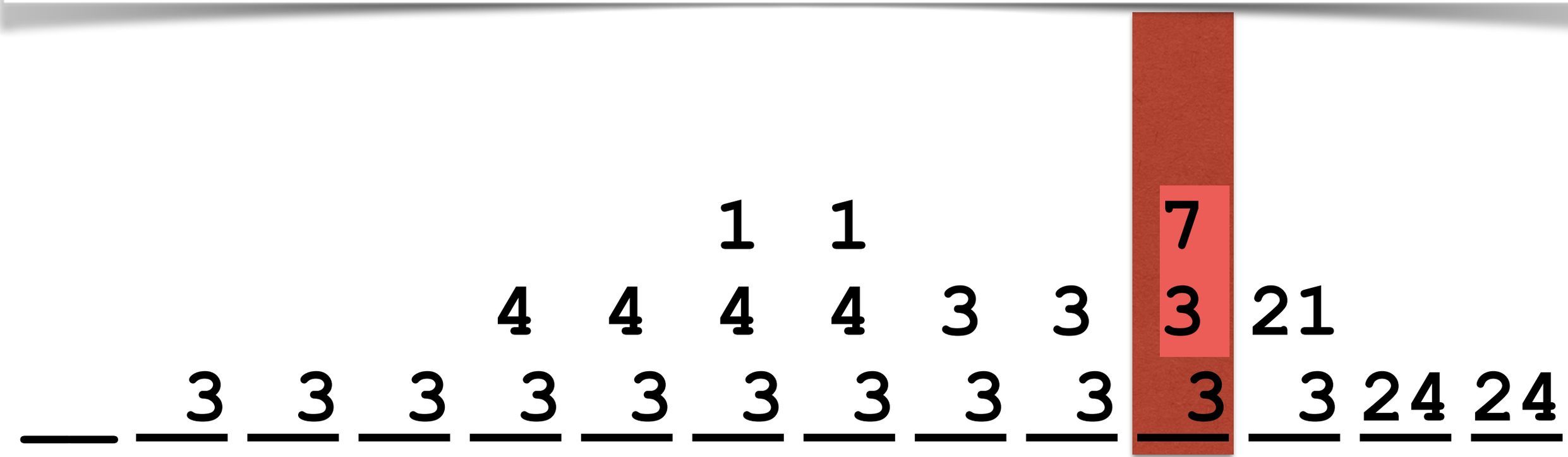

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```



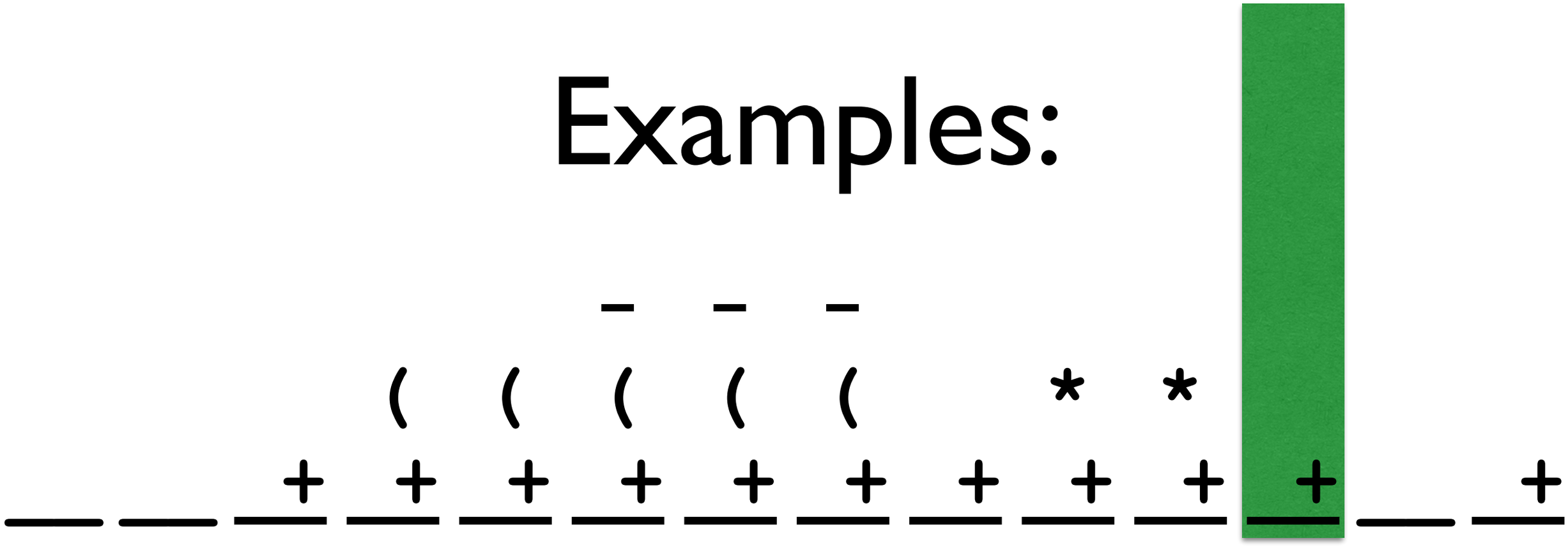
Examples:



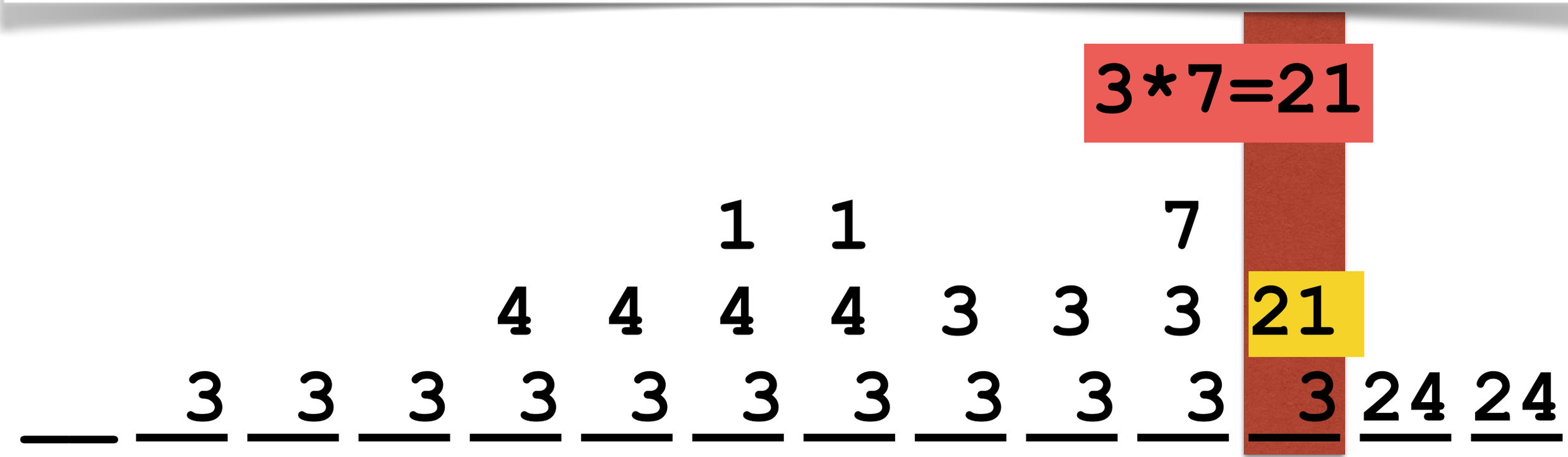

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```



Examples:




```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```



Examples:

						–	–	–						
				(	(	(	(	(		*	*			
		+	+	+	+	+	+	+	+	+	+			+
_	_	_	_	_	_	_	_	_	_	_	_	_	_	_

3	+	(	4	–	1	)	*	7	+	(	6	–	2	*

						1	1			7				
				4	4	4	4	3	3	3	21			
	3	3	3	3	3	3	3	3	3	3	3	24	24	
_	_	_	_	_	_	_	_	_	_	_	_	_	_	_

Examples:

					-	-	-						
			(	(	(	(	(			*	*		
		+	+	+	+	+	+	+	+	+	+		+
__	__	__	__	__	__	__	__	__	__	__	__	__	__

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

						1	1				7		
				4	4	4	4	3	3	3	21		
	3	3	3	3	3	3	3	3	3	3	3	24	24
__	__	__	__	__	__	__	__	__	__	__	__	__	__

3+21=24

Examples:

						-	-	-					
				(	(	(	(	(		*	*		
		+	+	+	+	+	+	+	+	+	+		+

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

						1	1			7			
				4	4	4	4	3	3	3	21		
	3	3	3	3	3	3	3	3	3	3	3	24	24

```
                                    +   +
                            (   (   (   (
                        *   *   *   *   *   *
                    -   -   -   -   -   -   -   -   -
            (   (   (   (   (   (   (   (   (   (   (
    +   +   +   +   +   +   +   +   +   +   +   +   +
```

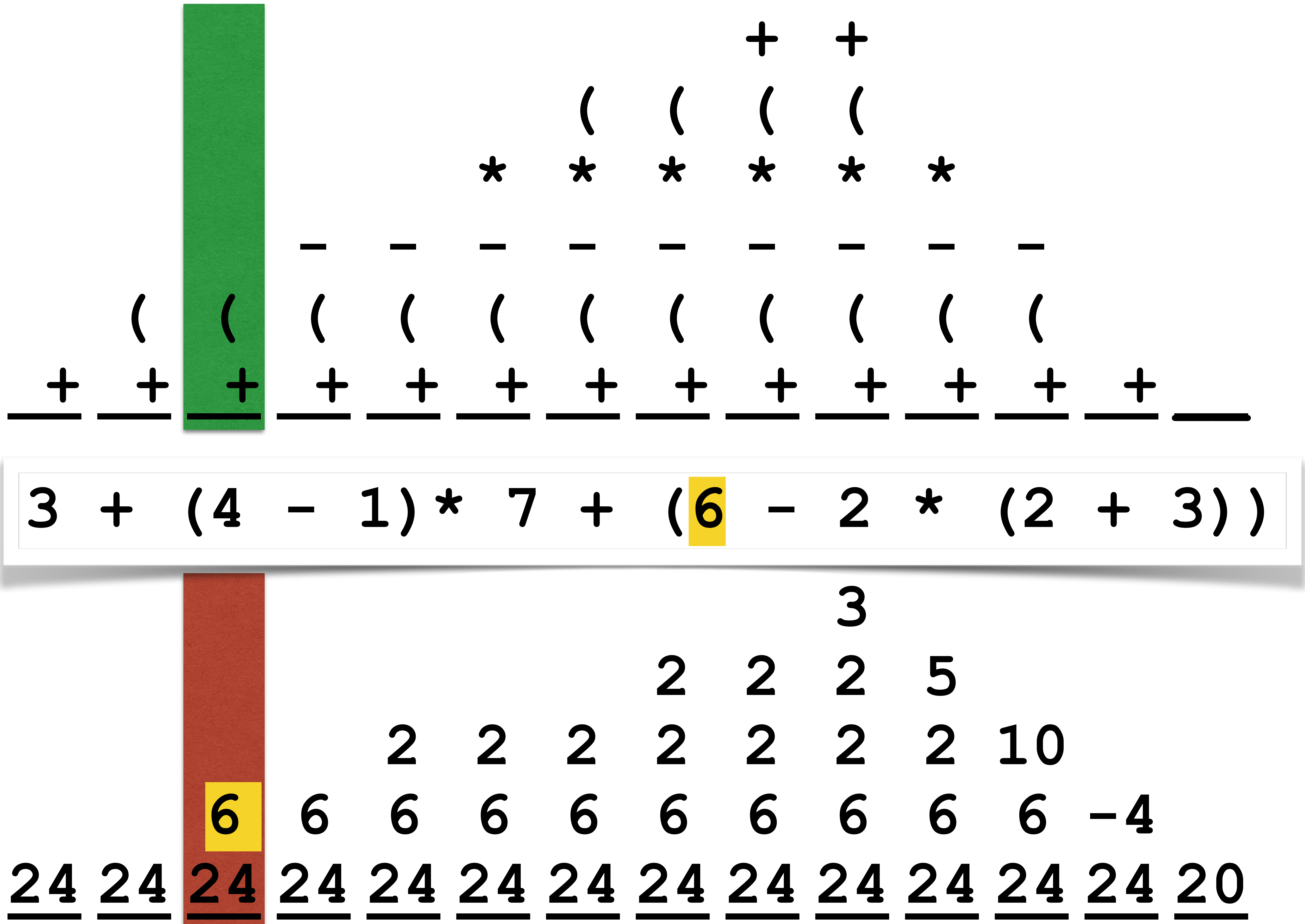
```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

```
                                3
                            2   2   2   5
                        2   2   2   2   2   2   10
                    6   6   6   6   6   6   6   6   6   -4
24  24  24  24  24  24  24  24  24  24  24  24  24  24  20
```

								+	+				
						(	(	(	(				
					*	*	*	*	*	*			
			-	-	-	-	-	-	-	-	-		
	(	(	(	(	(	(	(	(	(	(	(		
+	+	+	+	+	+	+	+	+	+	+	+	+	

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

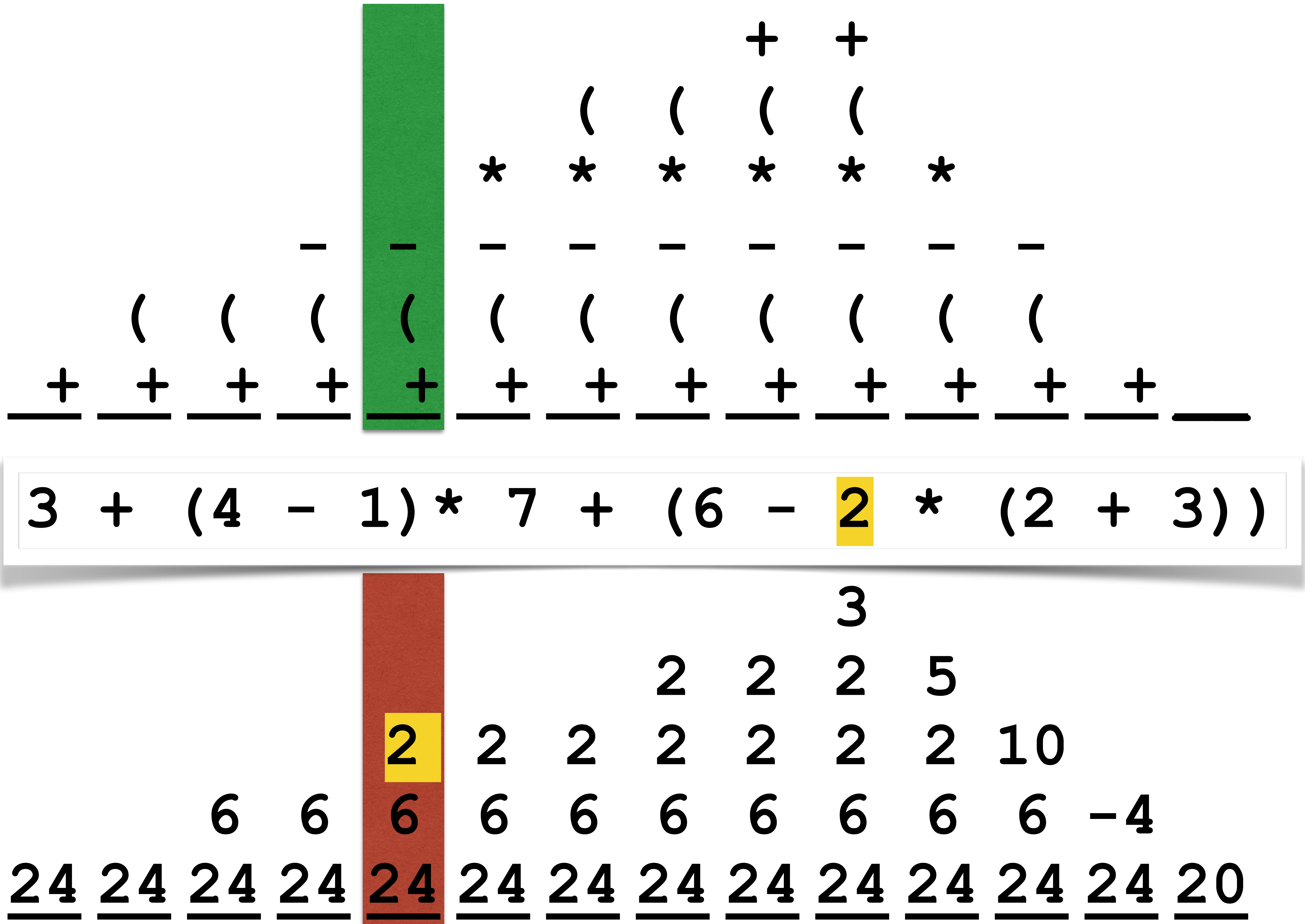
									3				
							2	2	2	5			
				2	2	2	2	2	2	2	10		
		6	6	6	6	6	6	6	6	6	6	-4	
24	24	24	24	24	24	24	24	24	24	24	24	24	20

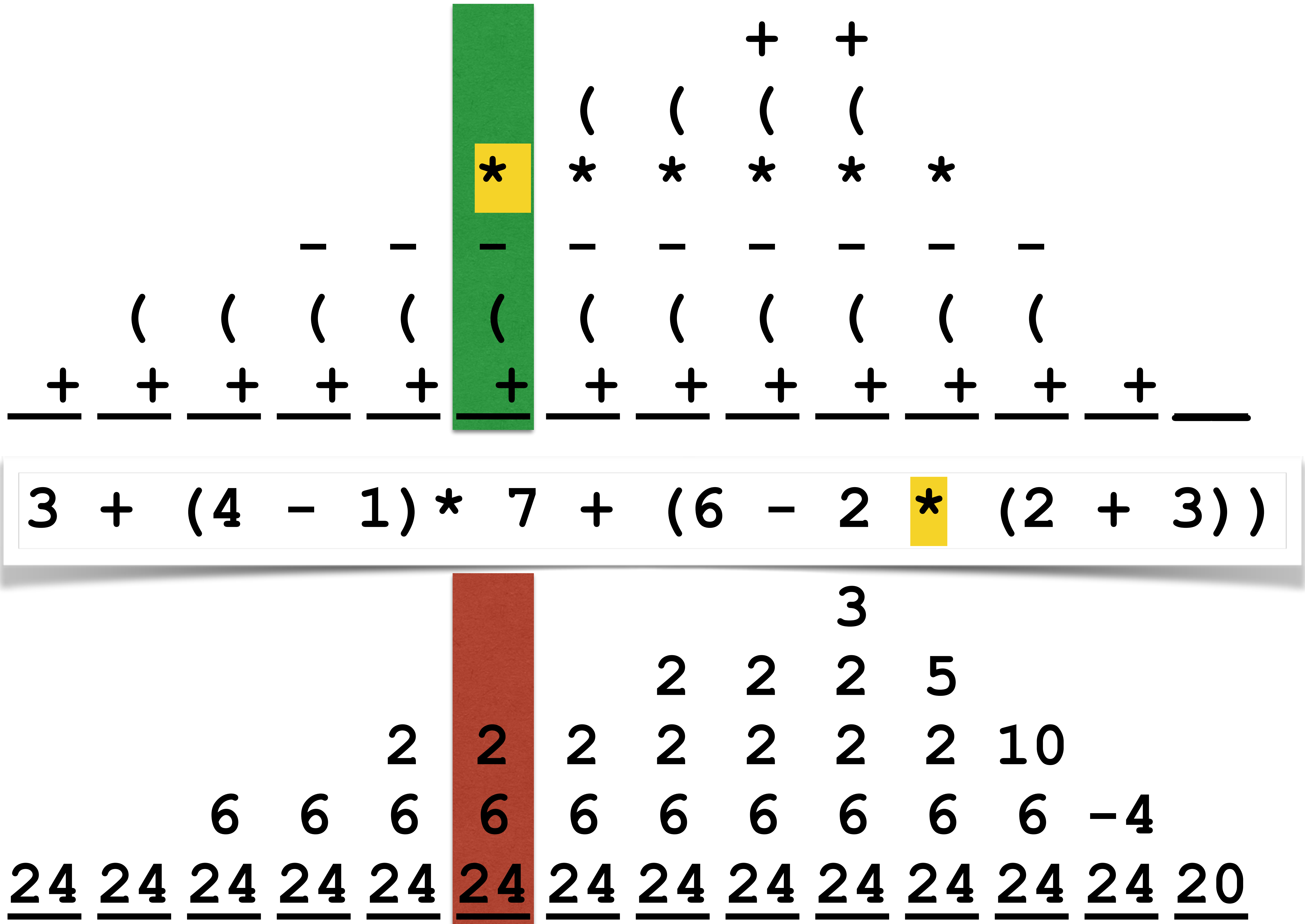

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

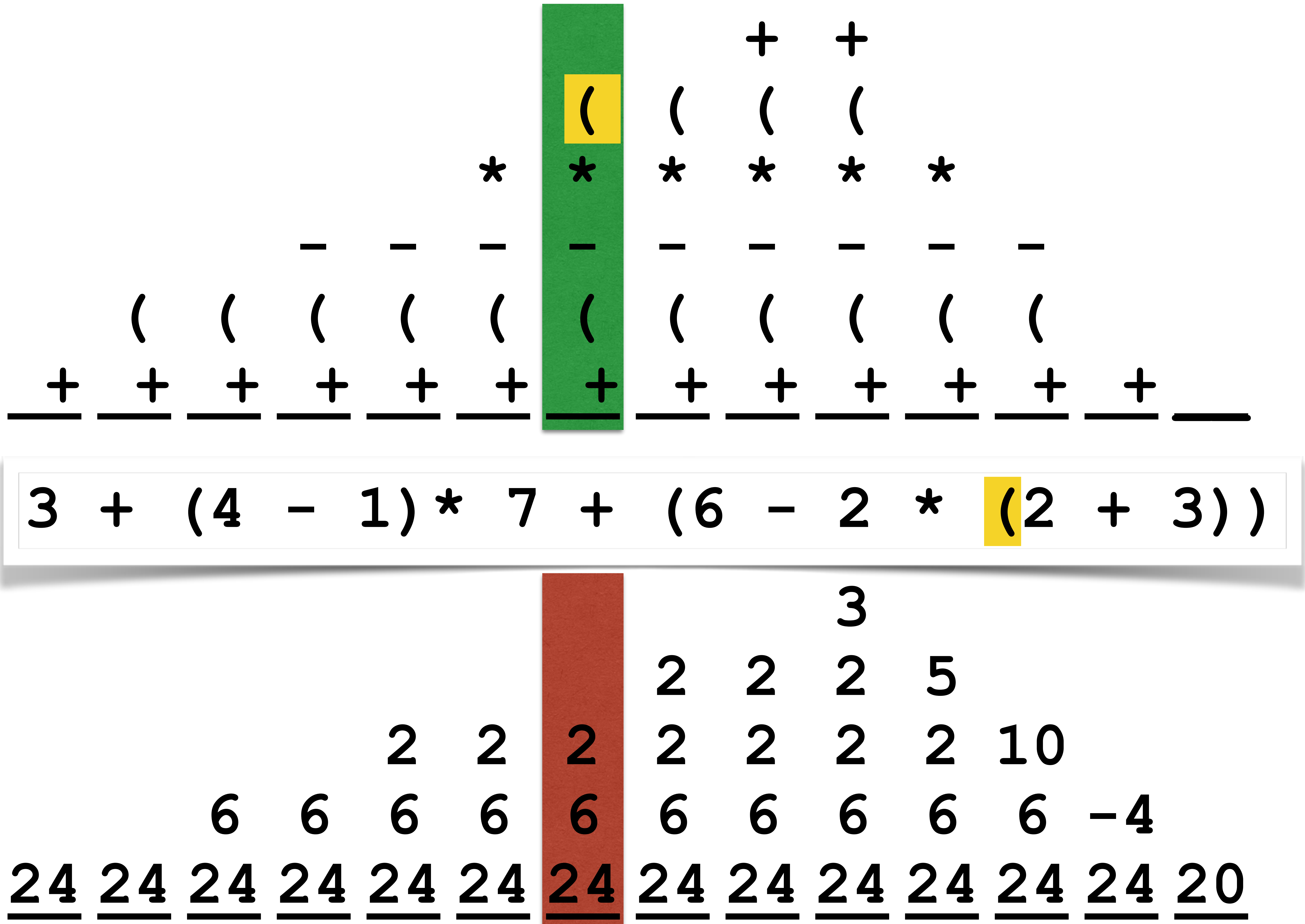
								+	+					
						(	(	(	(					
					*	*	*	*	*	*				
			−	−	−	−	−	−	−	−	−			
	(	(	(	(	(	(	(	(	(	(	(	(		
+	+	+	+	+	+	+	+	+	+	+	+	+	+	
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

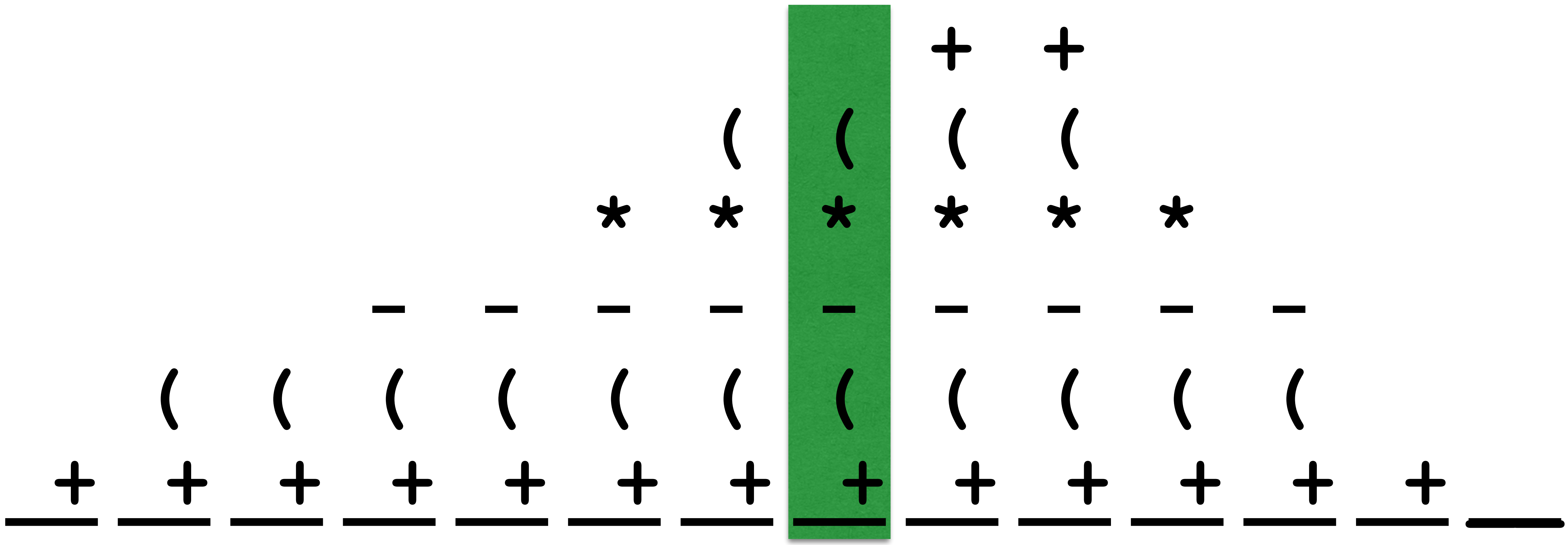
3 + (4 - 1)* 7 + (6 − 2 * (2 + 3))

									3					
						2	2	2	5					
				2	2	2	2	2	2	2	10			
		6	6	6	6	6	6	6	6	6	6	-4		
24	24	24	24	24	24	24	24	24	24	24	24	24	20	

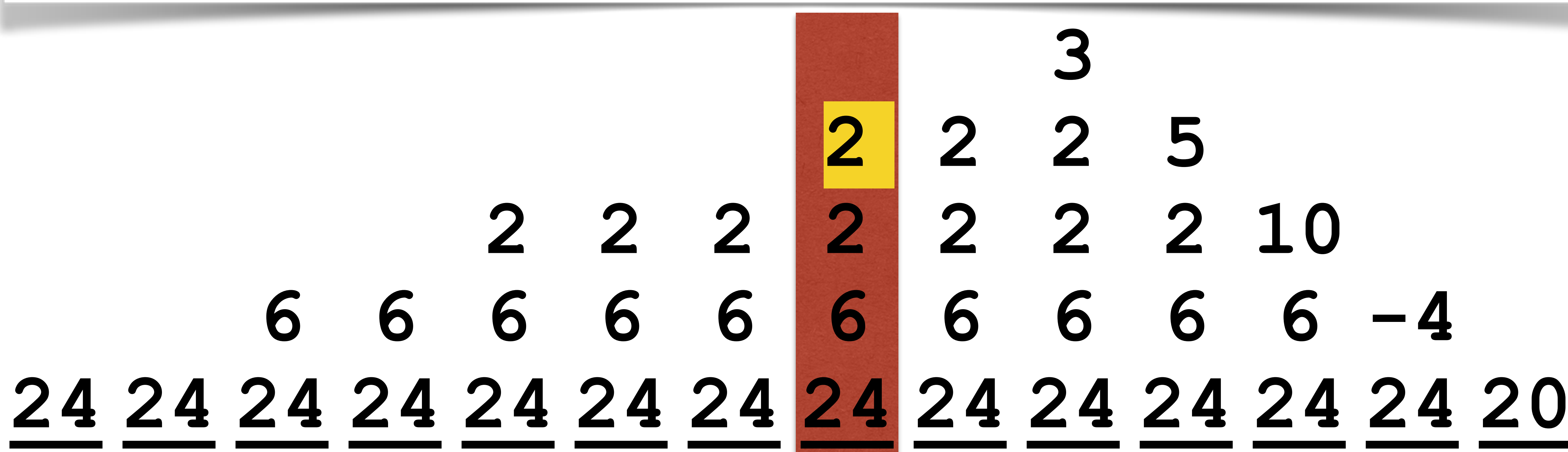

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

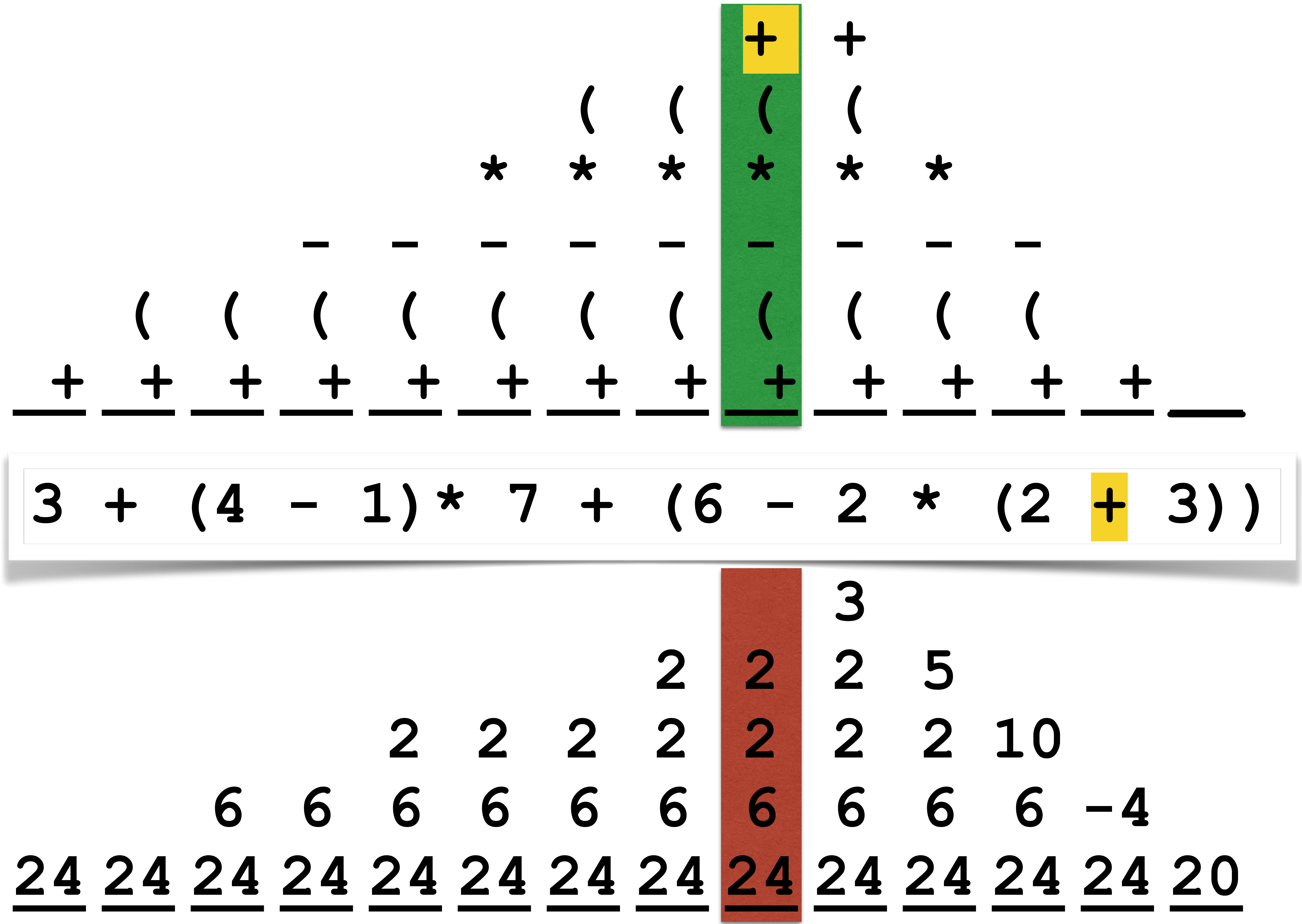

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))


3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))


+ +
((((
* * * * * *
- - - - - - - - -
(((((((((((
+ + + + + + + + + + + + +

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

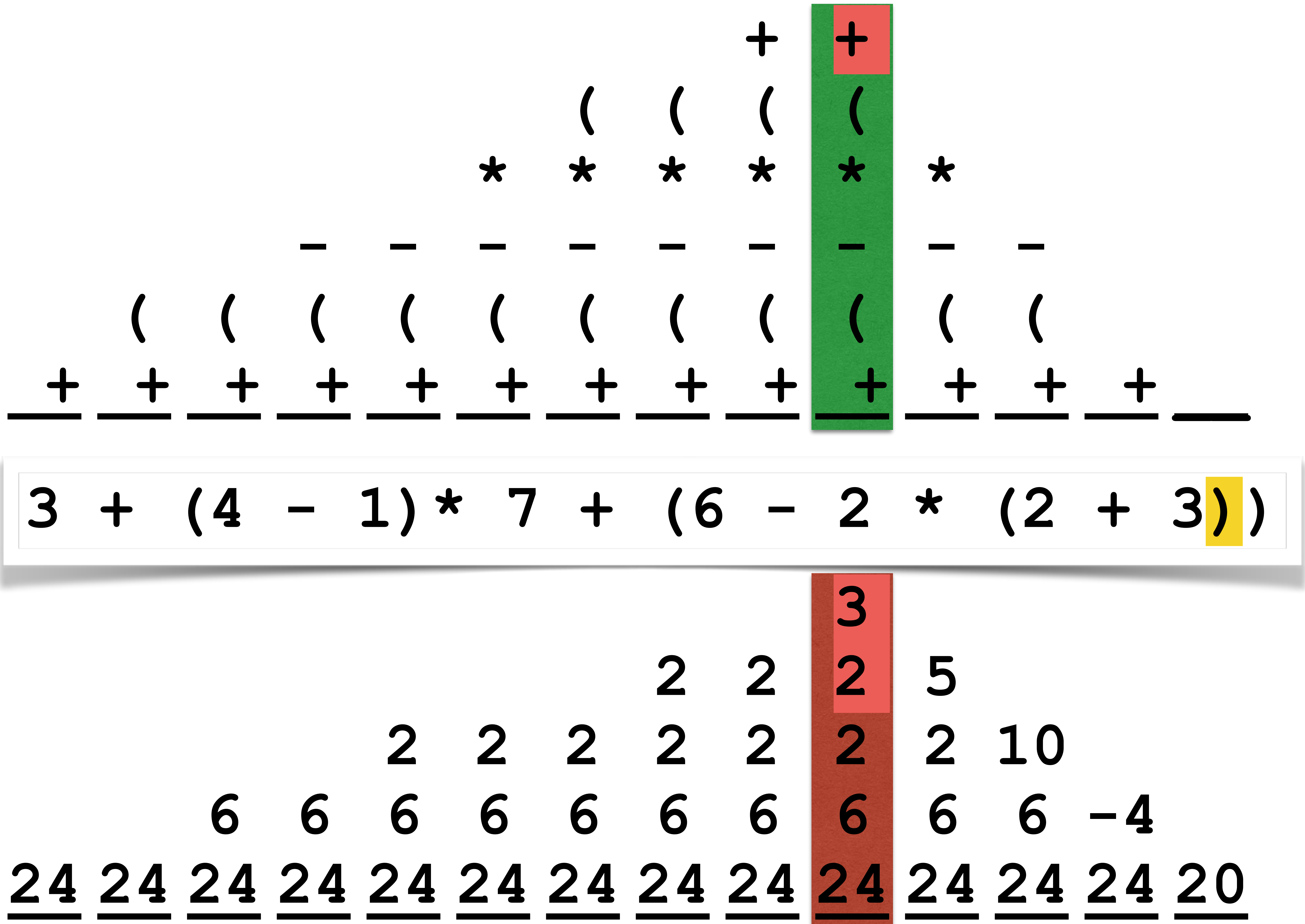

3
2 2 2 5
2 2 2 2 2 2 2 10
6 6 6 6 6 6 6 6 6 6 -4
24 24 24 24 24 24 24 24 24 24 24 24 24 20



+ +
((((
* * * * * *
- - - - - - - - -
(((((((((((
+ + + + + + + + + + + + +
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
3
2 2 2 5
2 2 2 2 2 2 2 10
6 6 6 6 6 6 6 6 6 6 -4
24 24 24 24 24 24 24 24 24 24 24 24 24 20

24	24	24	24	24	24	24	24	24	24	24	24	24	20
		6	6	6	6	6	6	6	6	6	6	-4	
				2	2	2	2	2	2	2	10		
						2	2	2	2	5			
									3				

3	+	(	(	4	-	1	)	*	7	+	(	6	-	2	*	(	2	+	3	)	)
	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	
		)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)		
		-	-	-	-	-	-	-	-	-											
			*	*	*	*	*	*													
				)	)	)	)														
				+	+																


3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

24	24	24	24	24	24	24	24	24	24	24	24	24	20
		6	6	6	6	6	6	6	6	6	6	-4	
				2	2	2	2	2	2	2	10		
						2	2	2	5				
						3	2+3=5						

	+	+	+	+	+	+	+	+	+	+	+	+	+
		)	)	)	)	)	)	)	)	)	)	)	
		-	-	-	-	-	-	-	-	-			
			*	*	*	*	*	*					
				)	)	)	)						
				+	+								

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

24	24	24	24	24	24	24	24	24	24	24	24	24	24	20
		6	6	6	6	6	6	6	6	6	6	6	-4	
				2	2	2	2	2	2	2	2	10		
						2	2	2	5					
							3							

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

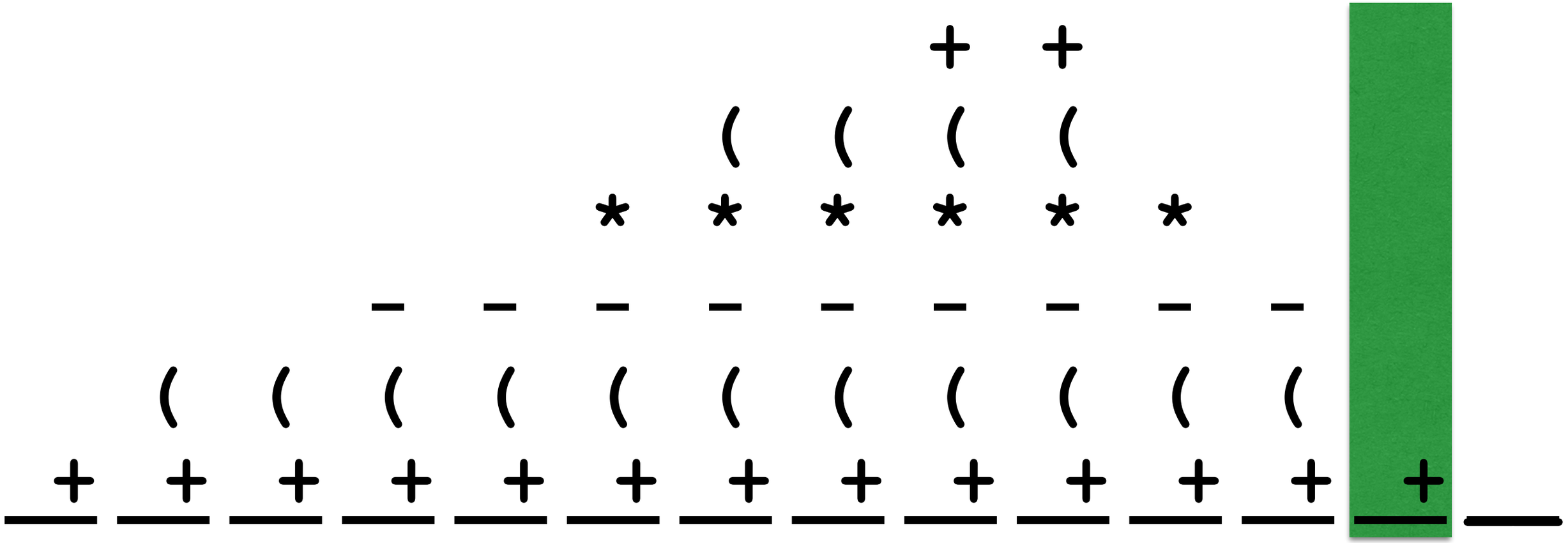
	+	+	+	+	+	+	+	+	+	+	+	+	+	+
		)	)	)	)	)	)	)	)	)	)	)	)	
		-	-	-	-	-	-	-	-	-				
			*	*	*	*	*							
				)	)	)	)							
				+	+									

24	24	24	24	24	24	24	24	24	24	24	24	24	20
		6	6	6	6	6	6	6	6	6	6	-4	
				2	2	2	2	2	2	2	10		
					2	2	2	5	2*5=10				
						3							

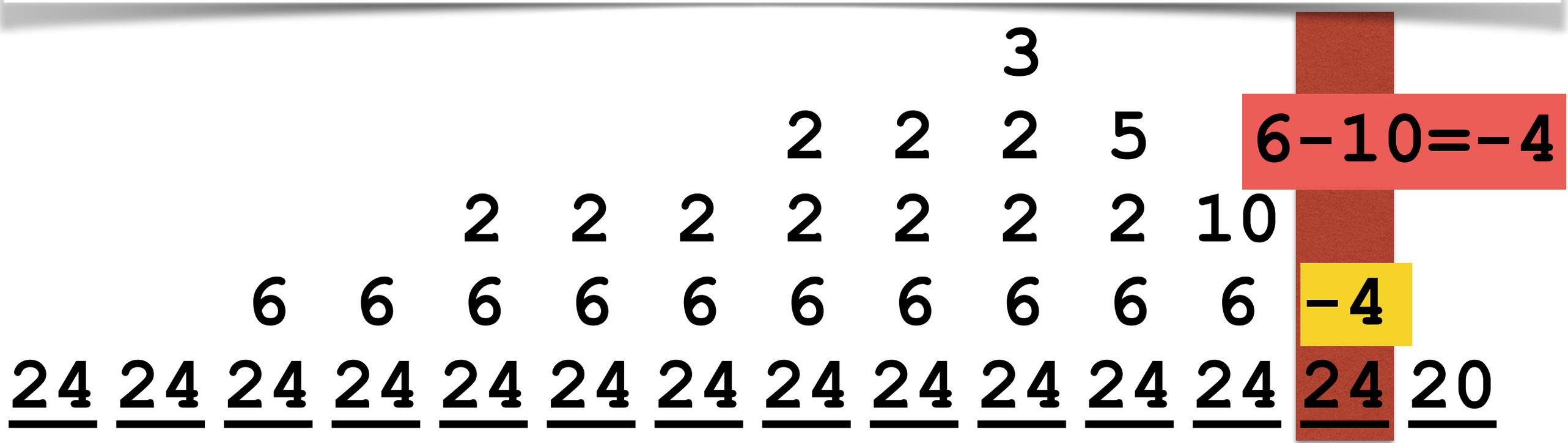
3	+	(	4	-	1	)	*	7	+	(	6	-	2	*	(	2	+	3	)
	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
		)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	)	
		-	-	-	-	-	-	-	-	-									
			*	*	*	*	*	*											
				)	)	)	)												
				+	+														

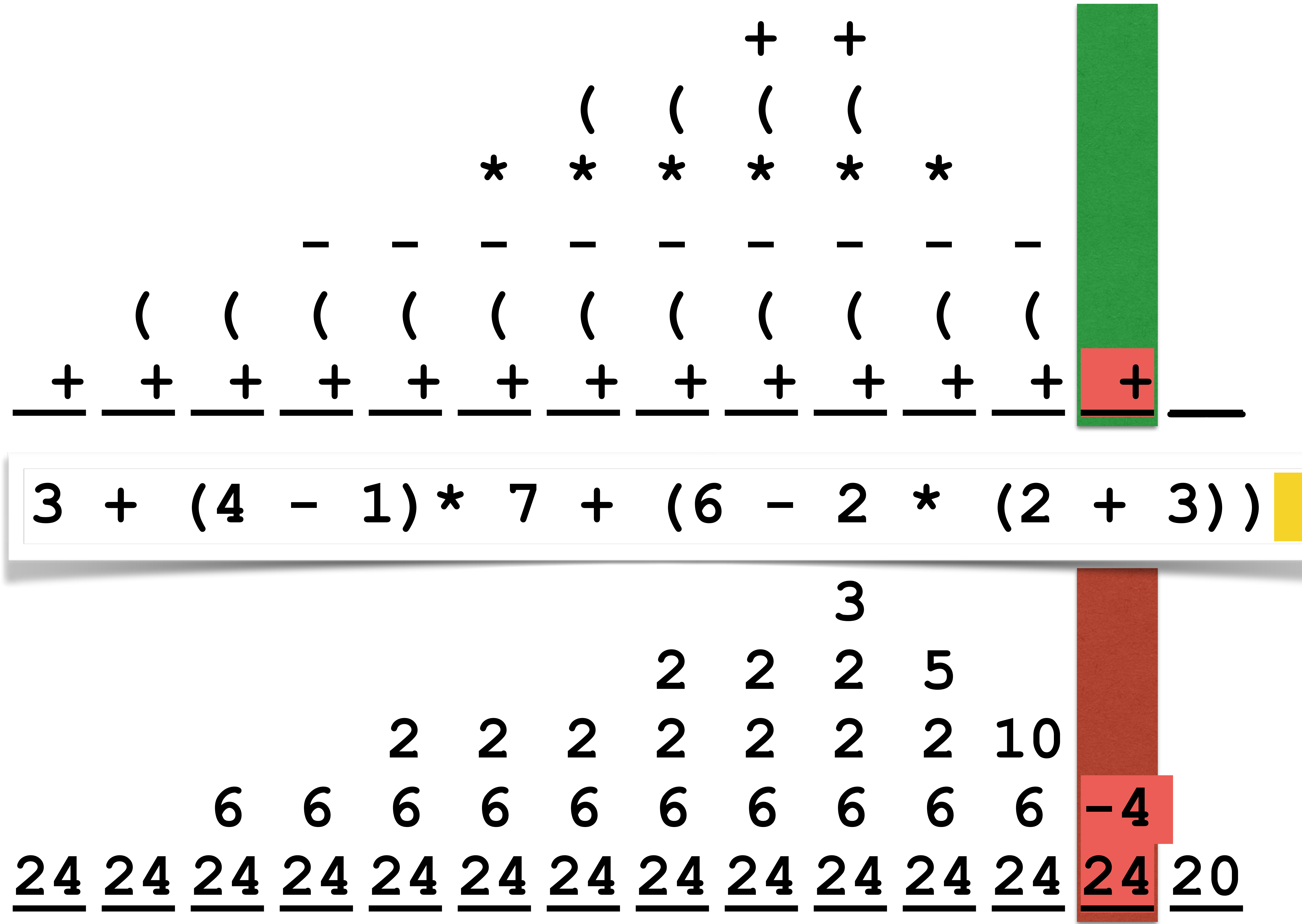
3	+	(	4	−	1	)	*	7	+	(	6	−	2	*
											3			
											2	2	5	
											2	2	2	10
											2	2	2	6
											6	6	6	−4
24	24	24	24	24	24	24	24	24	24	24	24	24	24	20

	+	+	+	+	+	+	+	+	+	+	+	+	+	+
		)	)	)	)	)	)	)	)	)	)	)	)	
		−	−	−	−	−	−	−	−					
			*	*	*	*	*	*						
				)	)	)	)							
				+	+									



3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))





3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
+ + + + + + + + + + + + +
(((((((((((
- - - - - - - - -
* * * * * *
((((
+ +
24 24 24 24 24 24 24 24 24 24 24 24 24 20
6 6 6 6 6 6 6 6 6 6 -4
2 2 2 2 2 2 2 10
2 2 2 5
3

20

24 24 24 24 24 24 24 24 24 24 24 24 24

6 6 6 6 6 6 6 6 6 6 6 6 -4

2 2 2 2 2 2 2 2 10

2 2 2 5

3

24-4=20

3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))

+ + + + + + + + + + + + +

)))))))))))

- - - - - - - - -

* * * * * *

))))

+ +

24	24	24	24	24	24	24	24	24	24	24	24	20
								6	6	6	6	-4
											10	6
											2	2
												2
												5
												2
												2
												2
												3

+	+	+	+	+	+	+	+	+	+	+	+	+
	)	)	)	)	)	)	)	)	)	)	)	
	−	−	−	−	−	−	−	−	−			
		*	*	*	*	*	*					
			)	)	)	)						
			+	+								

3 + (4 − 1)* 7 + (6 − 2 * (2 + 3))

Processing arithmetics

```
t=gettoken()
while type(t)≠eol do
   if type(t)=number then
   if type(t)=operator then
   if t="(" then
   if t=")" then
   t=gettoken()

while not isemptyO() do
   op=popO()
   arg2=popA()
   arg1=popA()
   pushA(exec(arg1,op,arg2))
return popA()
```

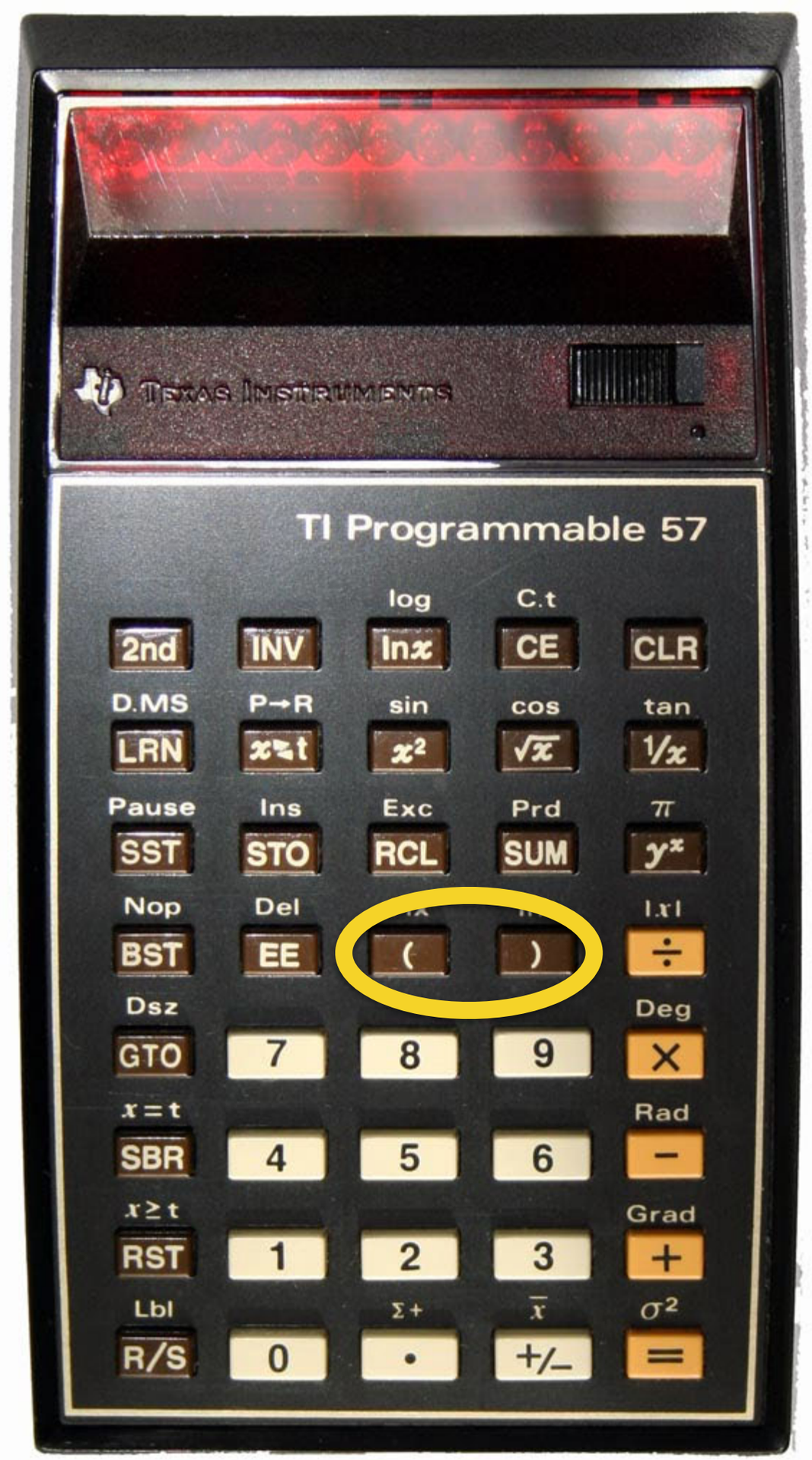
Processing arithmetics

```
if type(t)=number then pushA(t)

if type(t)=operator then
 if prio(t)≤prio(topO())
  then op=popO()
       arg2=popA()
       arg1=popA()
       pushA(exec(arg1,op,arg2))
pushO(t)
```

Processing arithmetics

```
if t="(" then pushO(t)

if t=")" then
    op=popO()
    while op≠"(" do
        arg2=popA()
        arg1=popA()
        pushA(exec(arg1,op,arg2))
        op=popO()
```

```
t=gettoken()
while type(t)≠eol do
  if type(t)=number then pushA(t)
  if type(t)=operator then
   if prio(t)≤prio(topO())
    then op=popO()
         arg2=popA()
         arg1=popA()
         pushA(exec(arg1,op,arg2))
     pushO(t)
  if t="(" then pushO(t)
  if t=")" then
     op=popO()
     while op≠"(" do
       arg2=popA()
       arg1=popA()
       pushA(exec(arg1,op,arg2))
       op=popO()
  t=gettoken()
while not isemptyO() do
  op=popO()
  arg2=popA()
  arg1=popA()
  pushA(exec(arg1,op,arg2))
return popA()
```


Texas Instruments
TI Programmable 57
2nd
INV
log
ln x
C.t
CE
CLR
D.MS
LRN
P→R
x≥t
sin
x²
cos
√x
tan
1/x
Pause
SST
Ins
STO
Exc
RCL
Prd
SUM
π
y^x
Nop
BST
Del
EE
(
)
|x|
÷
Dsz
GTO
7
8
9
Deg
×
x=t
SBR
4
5
6
Rad
−
x≥t
RST
1
2
3
Grad
+
Lbl
R/S
0
Σ+
•
x̄
+/−
σ²
=

TI vs HP


HP 50g
Graphing Calculator
EDIT
CURS
BIG
EVAL
FACTO
SIMP
Y=
WIN
GRAPH
2D/3D
TBLSET
TABLE
F1 A
F2 B
F3 C
F4 D
F5 E
F6 F
FILES BEGIN CUSTOM END i L
APPS G
MODE H
TOOL I
UPDIR COPY RCL CUT PREV PASTE
VAR J
STO K
NXT L
CMD UNDO PRG CHARS MTRW EQW MTH CAT DEL CLEAR
HIST M
EVAL N
' O
SYMB P
e^x LN x² ∜y ASIN Σ ACOS ∂ ATAN ∫
y^x Q
√x R
SIN S
COS T
TAN U
10^x LOG ≠ = ≤ < ≥ > ABS ARG
EEX V
+/− W
X X
1/X Y
÷ Z
USER ENTRY S.SLV NUM.SLV EXP&LN TRIG FINANCE TIME [] ""
ALPHA
7
8
9
×
CALC ALG MATRICES STAT CONVERT UNITS () −
4
5
6
−
ARITH CMPLX DEF LIB # BASE {} ≪≫
1
2
3
+
CONT OFF ∞ → :: ↵ π , ANS − NUM
ON
0
•
SPC
ENTER
CANCEL

Examples:

```
3 + (4 - 1)* 7 + (6 - 2 * (2 + 3))
```

```
3 4 1 - 7 * + 6 2 2 3 + * - +
```

```
                                3
                              2 2 5
          1     7           2 2 2 2 10
      4 4 3 3 21     6 6 6 6 6 6 -4
  3 3 3 3 3 3 24 24 24 24 24 24 24 20
```

Examples:

```
3 4 1 - 7 * + 6 2 2 3 + * - +
```

```
                                3
                              2 2 5
          1     7           2 2 2 2 10
      4 4 3 3 21     6 6 6 6 6 6 -4
    3 3 3 3 3 3 2 4 2 4 2 4 2 4 2 4 2 0
```

Examples:

3	4	1	-	7	*	+	6	2	2	3	+	*	-	+
										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

3	4	1	-	7	*	+	6	2	2	3	+	*	-	+
										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

3	4	1	-	7	*	+	6	2	2	3	+	*	-	+
										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	2	4	2	4	2	4	2	4	20

Examples:

3 4 1 - 7 * + 6 2 2 3 + * - +

															3							
													2	2	5							
			1			7						2	2	2	2	10						
		4	4		3	3	21			6	6	6	6	6	6	-4						
	3	3	3	3	3	3	2	4	2	4	2	4	2	4	2	4	2	4	2	0		

Examples:

3 4 1 - 7 * + 6 2 2 3 + * - +

															3							
													2	2	5							
			1			7						2	2	2	2	10						
		4	4	3		3	21			6	6	6	6	6	6	-4						
	3	3	3	3	3	3	2	4	2	4	2	4	2	4	2	4	2	4	2	4	2	0

Examples:

3 4 1 - 7 * + 6 2 2 3 + * - +

```
                                  3
                              2   2   5
          1       7       2   2   2   2  10
      4   4   3   3  21       6   6   6   6   6   6  -4
  3   3   3   3   3   3  24  24  24  24  24  24  24  24  20
```

Examples:

3 4 1 - 7 * + 6 2 2 3 + * - +

												3						
										2	2	5						
		1		7					2	2	2	2	10					
	4	4	3	3	21		6	6	6	6	6	6	-4					
3	3	3	3	3	3	2	4	2	4	2	4	2	4	2	4	2	4	20

Examples:

```
3 4 1 - 7 * + 6 2 2 3 + * - +
```

```
                                3
                            2   2   5
        1       7           2   2   2   2  10
    4   4   3   3  21       6   6   6   6   6   6  -4
  3   3   3   3   3   3  2  4  2  4  2  4  2  4  2  4  2  0
```

Examples:

```
3 4 1 - 7 * + 6 2 2 3 + * - +
```

```
                                3
                                2 2 5
          1     7               2 2 2 2 10
    4 4 3 3 21      6 6 6 6 6 6 -4
  3 3 3 3 3 3 2 4 2 4 2 4 2 4 2 4 2 4 2 0
```

Examples:

3	4	1	-	7	*	+	6	2	2	3	+	*	-	+

										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

`3 4 1 - 7 * + 6 2 2 3 + * - +`

											3				
										2	2	5			
			1		7				2	2	2	2	10		
		4	4	3	3	21		6	6	6	6	6	6	-4	
	3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

```
3 4 1 - 7 * + 6 2 2 3 + * - +
```

										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

3	4	1	-	7	*	+	6	2	2	3	+	*	-	+
										3				
									2	2	5			
		1		7				2	2	2	2	10		
	4	4	3	3	21		6	6	6	6	6	6	-4	
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20

Examples:

3 4 1 - 7 * + 6 2 2 3 + * - +

										3							
								2	2	5							
		1		7				2	2	2	2	10					
	4	4	3	3	21		6	6	6	6	6	6	-4				
3	3	3	3	3	3	24	24	24	24	24	24	24	24	20			

Winter 2016
COMP-250: Introduction to Computer Science

Lecture 7, February 2, 2016