Notes: Data Analysis in Spreadsheets

Created by Wenxiao Zhou

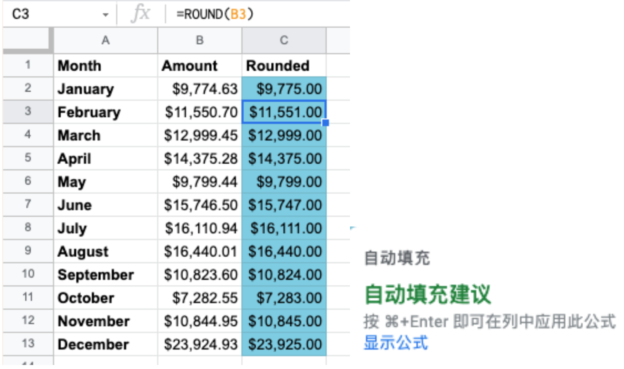
1.Predefined Functions

(1) First function: ROUND

Functions perform calculations on your data. For example, the ROUND function calculates the rounded value of its input.

ROUND(value): rounds the number you give as input, value.

Automatically calculate values by ‘command+return’ or drag down the right corner

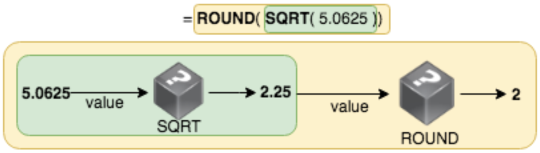


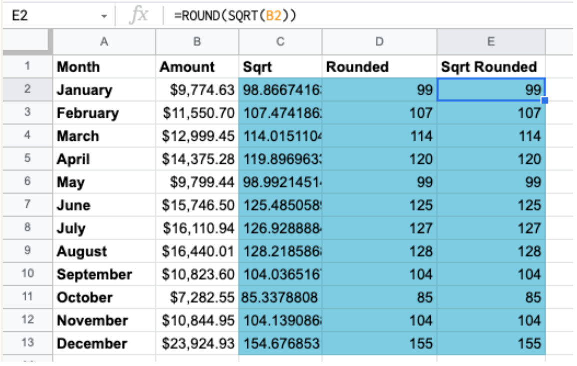
C3 =ROUND(B3)

	A	B	C
1	Month	Amount	Rounded
2	January	$9,774.63	$9,775.00
3	February	$11,550.70	$11,551.00
4	March	$12,999.45	$12,999.00
5	April	$14,375.28	$14,375.00
6	May	$9,799.44	$9,799.00
7	June	$15,746.50	$15,747.00
8	July	$16,110.94	$16,111.00
9	August	$16,440.01	$16,440.00
10	September	$10,823.60	$10,824.00
11	October	$7,282.55	$7,283.00
12	November	$10,844.95	$10,845.00
13	December	$23,924.93	$23,925.00

(2) Function composition-SQRT

Google Sheets will first evaluate the innermost function and use the result as an argument for the outer function. Combining functions like this is called function composition.





E2 =ROUND(SQRT(B2))

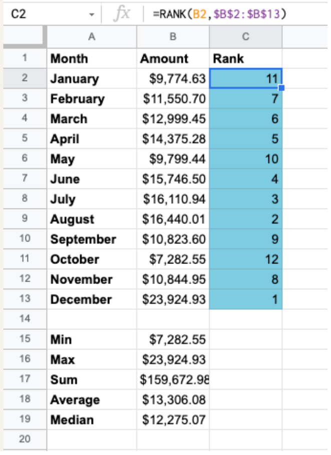
	A	B	C	D	E
1	Month	Amount	Sqrt	Rounded	Sqrt Rounded
2	January	$9,774.63	98.8667416	99	99
3	February	$11,550.70	107.474186	107	107
4	March	$12,999.45	114.015110	114	114
5	April	$14,375.28	119.896963	120	120
6	May	$9,799.44	98.9921451	99	99
7	June	$15,746.50	125.485058	125	125
8	July	$16,110.94	126.928888	127	127
9	August	$16,440.01	128.218586	128	128
10	September	$10,823.60	104.036516	104	104
11	October	$7,282.55	85.3378808	85	85
12	November	$10,844.95	104.139086	104	104
13	December	$23,924.93	154.676853	155	155

(3) Functions and ranges-MIN, MAX

Arguments can be ranges, where every value in the range is checked (e.g. =MAX(A1:A7)):
MIN(value1, [value2, ...]): searches for the minimum value in its arguments
MAX(value1, [value2, ...]): searches for the maximum value in its arguments

(4) Selecting ranges- SUM, AVERAGE, MEDIAN
SUM(value1, [value2, ...]): calculates the sum of all its arguments
AVERAGE(value1, [value2, ...]): calculates the average of all its arguments
MEDIAN(value1, [value2, ...]): calculates the median of all its arguments

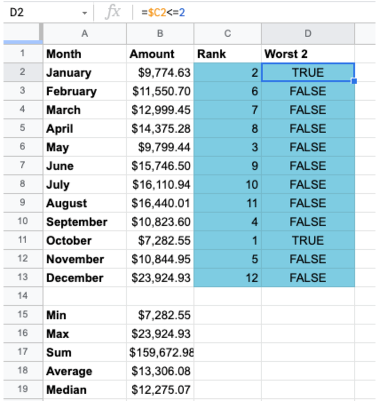
(5) Multiple arguments – RANK
RANK gives you an idea how a value compares to other values in a range.
RANK(value, data): evaluates to the rank of value in a range, data



C2 =RANK(B2,$B$2:$B$13)

	A	B	C
1	Month	Amount	Rank
2	January	$9,774.63	11
3	February	$11,550.70	7
4	March	$12,999.45	6
5	April	$14,375.28	5
6	May	$9,799.44	10
7	June	$15,746.50	4
8	July	$16,110.94	3
9	August	$16,440.01	2
10	September	$10,823.60	9
11	October	$7,282.55	12
12	November	$10,844.95	8
13	December	$23,924.93	1
14			
15	Min	$7,282.55	
16	Max	$23,924.93	
17	Sum	$159,672.98	
18	Average	$13,306.08	
19	Median	$12,275.07	
20			

More arguments of RANK:
This time, use the third argument, is_ascending, to get the rank of the value where the data list is considered in an ascending order.
RANK(value, data, [is_ascending]): when is_ascending is 1, the rank is considered in an ascending order of the data. It defaults to 0, meaning the rank will be considered in a descending list of data. See the table below for an example.
Find the worst 2 months in amount counts:



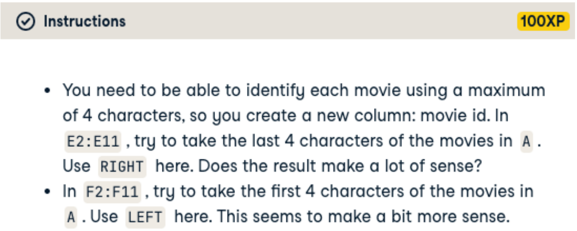
D2 =$C2<=2

	A	B	C	D
1	Month	Amount	Rank	Worst 2
2	January	$9,774.63	2	TRUE
3	February	$11,550.70	6	FALSE
4	March	$12,999.45	7	FALSE
5	April	$14,375.28	8	FALSE
6	May	$9,799.44	3	FALSE
7	June	$15,746.50	9	FALSE
8	July	$16,110.94	10	FALSE
9	August	$16,440.01	11	FALSE
10	September	$10,823.60	4	FALSE
11	October	$7,282.55	1	TRUE
12	November	$10,844.95	5	FALSE
13	December	$23,924.93	12	FALSE
14				
15	Min	$7,282.55		
16	Max	$23,924.93		
17	Sum	$159,672.98		
18	Average	$13,306.08		
19	Median	$12,275.07		

(6) String manipulation – LEFT, RIGHT

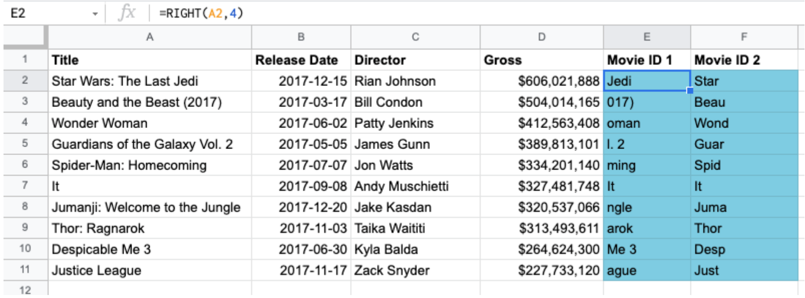
LEFT(string, [number_of_characters]): selects the leftmost part of a string. The number of characters selected is defined in the optional argument number_of_characters, and defaults to 1.

RIGHT(string, [number_of_characters]): selects the rightmost part of a string. The number of characters selected is defined in the optional argument number_of_characters, and defaults to 1.



Instructions 100XP

- You need to be able to identify each movie using a maximum of 4 characters, so you create a new column: movie id. In `E2:E11`, try to take the last 4 characters of the movies in `A`. Use `RIGHT` here. Does the result make a lot of sense?
- In `F2:F11`, try to take the first 4 characters of the movies in `A`. Use `LEFT` here. This seems to make a bit more sense.



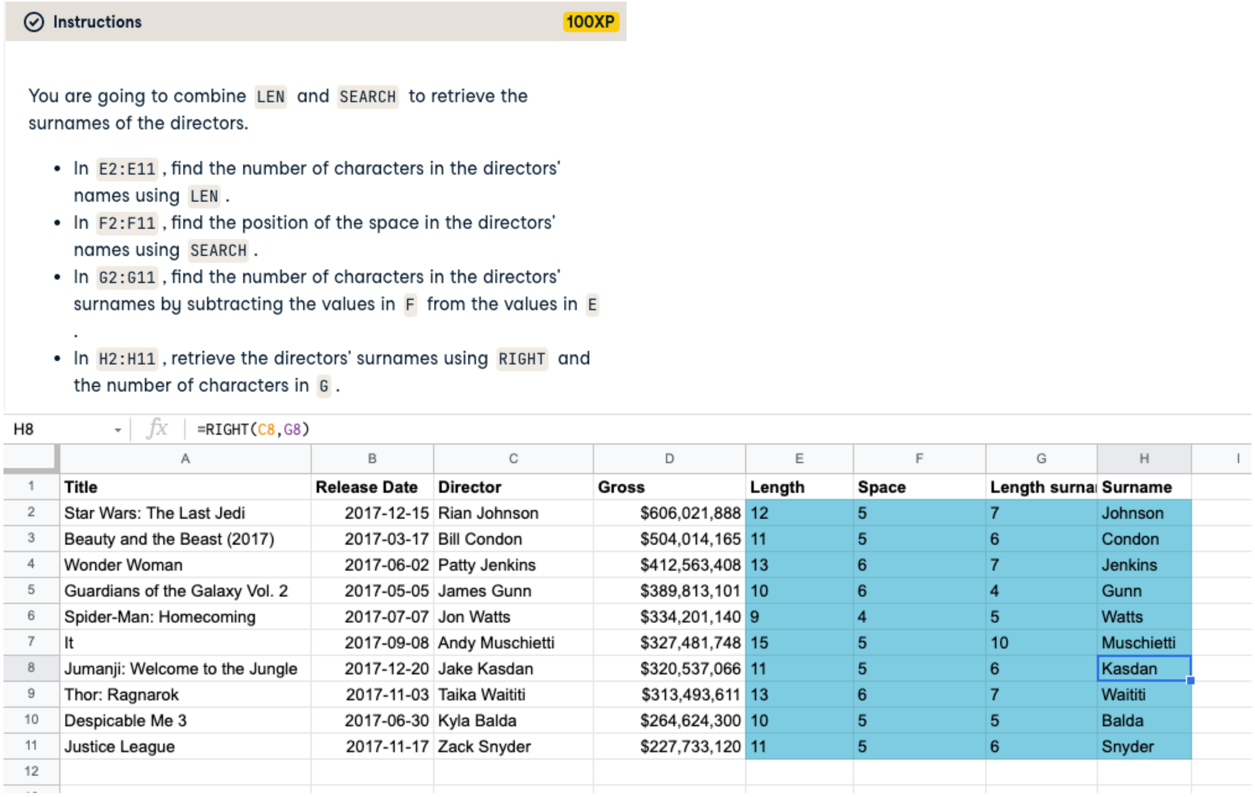
E2 =RIGHT(A2,4)

	A	B	C	D	E	F
1	Title	Release Date	Director	Gross	Movie ID 1	Movie ID 2
2	Star Wars: The Last Jedi	2017-12-15	Rian Johnson	$606,021,888	Jedi	Star
3	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	017)	Beau
4	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	oman	Wond
5	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	l. 2	Guar
6	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	ming	Spid
7	It	2017-09-08	Andy Muschietti	$327,481,748	It	It
8	Jumanji: Welcome to the Jungle	2017-12-20	Jake Kasdan	$320,537,066	ngle	Juma
9	Thor: Ragnarok	2017-11-03	Taika Waititi	$313,493,611	arok	Thor
10	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	Me 3	Desp
11	Justice League	2017-11-17	Zack Snyder	$227,733,120	ague	Just
12						

(7) String Information- LEN, SEARCH

LEN(text): evaluates to the number of characters of text. E.g. =LEN("Cell") would evaluate to 4.

SEARCH(search_for, text_to_search): searches for search_for in text_to_search:
search_for: the string to look for
text_to_search: the string to look in
SEARCH evaluates to a number, the location in the string where search_for appears, with 1 being the first character. E.g. =SEARCH("e", "test test") would evaluate to 2, because the first "e" appears as the second character.



Instructions 100XP

You are going to combine `LEN` and `SEARCH` to retrieve the surnames of the directors.

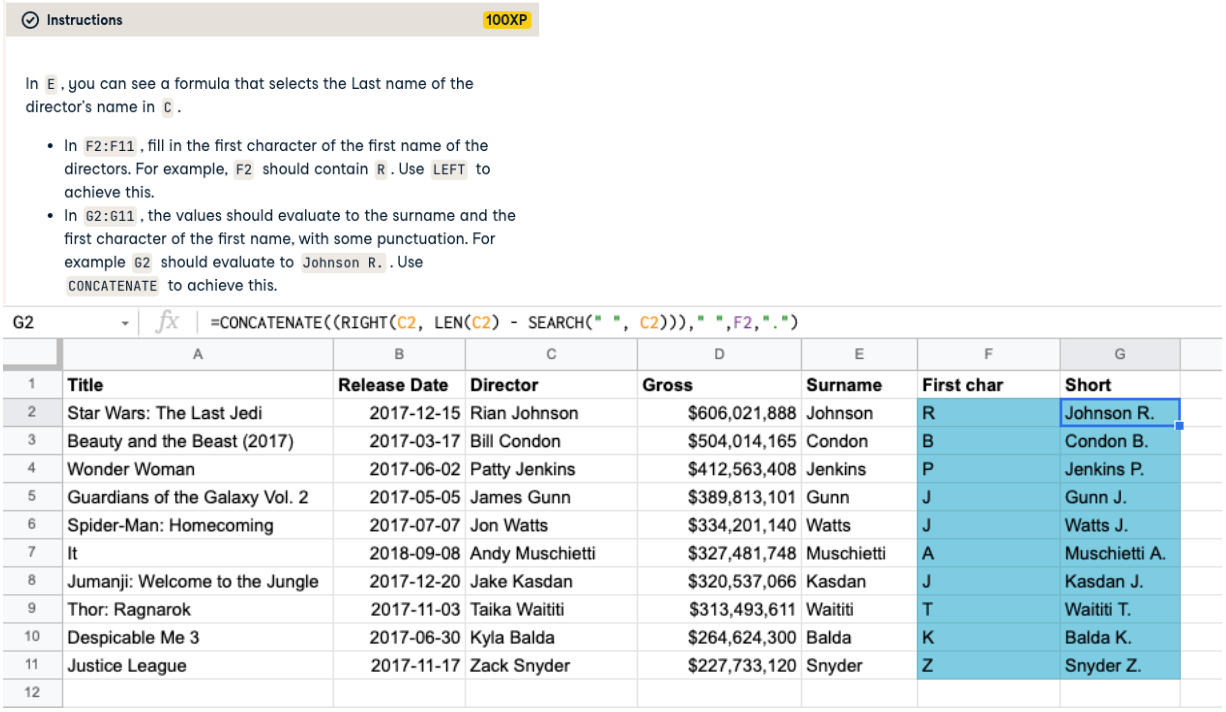
- In `E2:E11`, find the number of characters in the directors' names using `LEN`.
- In `F2:F11`, find the position of the space in the directors' names using `SEARCH`.
- In `G2:G11`, find the number of characters in the directors' surnames by subtracting the values in `F` from the values in `E`
  .
- In `H2:H11`, retrieve the directors' surnames using `RIGHT` and the number of characters in `G`.

```
H8    =RIGHT(C8,G8)
```

	A	B	C	D	E	F	G	H
1	Title	Release Date	Director	Gross	Length	Space	Length surna	Surname
2	Star Wars: The Last Jedi	2017-12-15	Rian Johnson	$606,021,888	12	5	7	Johnson
3	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	11	5	6	Condon
4	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	13	6	7	Jenkins
5	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	10	6	4	Gunn
6	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	9	4	5	Watts
7	It	2017-09-08	Andy Muschietti	$327,481,748	15	5	10	Muschietti
8	Jumanji: Welcome to the Jungle	2017-12-20	Jake Kasdan	$320,537,066	11	5	6	Kasdan
9	Thor: Ragnarok	2017-11-03	Taika Waititi	$313,493,611	13	6	7	Waititi
10	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	10	5	5	Balda
11	Justice League	2017-11-17	Zack Snyder	$227,733,120	11	5	6	Snyder
12								

(8) Combining strings – CONCATENATE

CONCATENATE(string1, [string2, ...]): combines one or more strings into a single string. E.g. =CONCATENATE("foo", " ", "bar") evaluates to foo bar



Instructions 100XP

In `E`, you can see a formula that selects the Last name of the director's name in `C`.

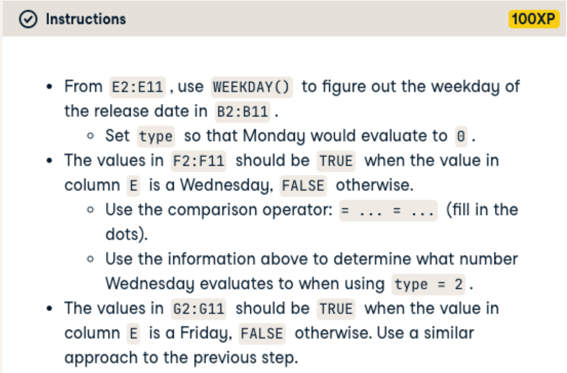
- In `F2:F11`, fill in the first character of the first name of the directors. For example, `F2` should contain `R`. Use `LEFT` to achieve this.
- In `G2:G11`, the values should evaluate to the surname and the first character of the first name, with some punctuation. For example `G2` should evaluate to `Johnson R.`. Use `CONCATENATE` to achieve this.

```
G2    =CONCATENATE((RIGHT(C2, LEN(C2) - SEARCH(" ", C2)))," ",F2,".")
```

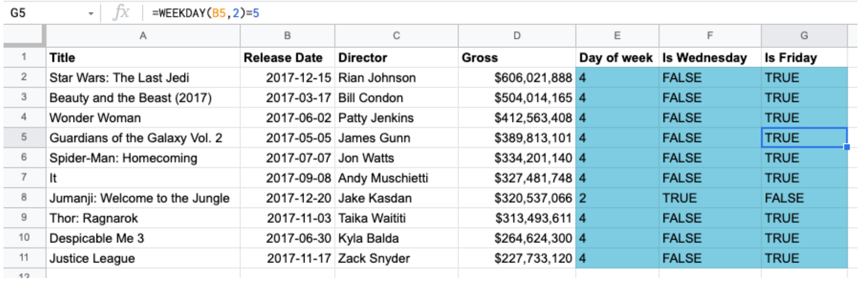
	A	B	C	D	E	F	G
1	Title	Release Date	Director	Gross	Surname	First char	Short
2	Star Wars: The Last Jedi	2017-12-15	Rian Johnson	$606,021,888	Johnson	R	Johnson R.
3	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	Condon	B	Condon B.
4	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	Jenkins	P	Jenkins P.
5	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	Gunn	J	Gunn J.
6	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	Watts	J	Watts J.
7	It	2018-09-08	Andy Muschietti	$327,481,748	Muschietti	A	Muschietti A.
8	Jumanji: Welcome to the Jungle	2017-12-20	Jake Kasdan	$320,537,066	Kasdan	J	Kasdan J.
9	Thor: Ragnarok	2017-11-03	Taika Waititi	$313,493,611	Waititi	T	Waititi T.
10	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	Balda	K	Balda K.
11	Justice League	2017-11-17	Zack Snyder	$227,733,120	Snyder	Z	Snyder Z.
12							

(9) Date Functions – WEEKDAY

WEEKDAY(date, [type]): evaluates to the day of the week of a date. type is 1, 2 or 3.
type = 1: Sunday is day 1 and Saturday is day 7 (default)
type = 2: Monday is day 1 and Sunday is day 7
type = 3: Monday is day 0 and Sunday is day 6
For example, using =WEEKDAY(A1, 2) (where A1 contains the date 2019-01-01) would evaluate to 2, because January 1st 2019 fell on a Tuesday and setting type to 2 sets Monday at 1.



- From `E2:E11`, use `WEEKDAY()` to figure out the weekday of the release date in `B2:B11`.
  - Set `type` so that Monday would evaluate to `0`.
- The values in `F2:F11` should be `TRUE` when the value in column `E` is a Wednesday, `FALSE` otherwise.
  - Use the comparison operator: `= ... = ...` (fill in the dots).
  - Use the information above to determine what number Wednesday evaluates to when using `type = 2`.
- The values in `G2:G11` should be `TRUE` when the value in column `E` is a Friday, `FALSE` otherwise. Use a similar approach to the previous step.

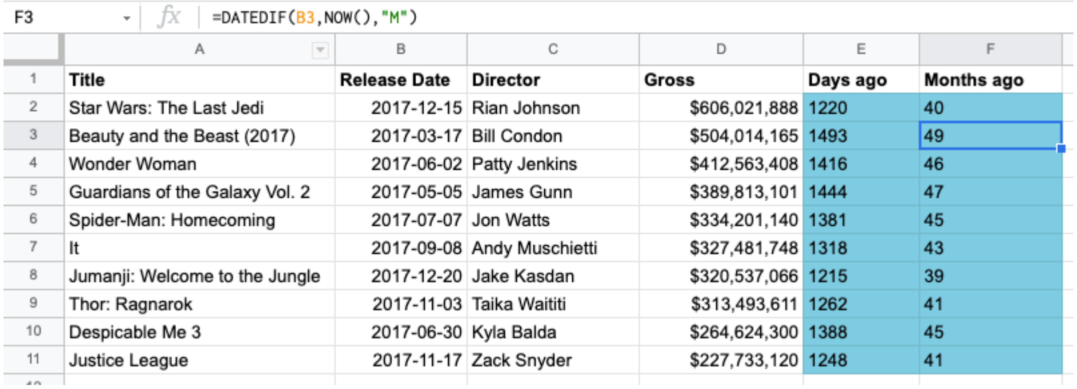


```
=WEEKDAY(B5,2)=5
```

	A	B	C	D	E	F	G
1	Title	Release Date	Director	Gross	Day of week	Is Wednesday	Is Friday
2	Star Wars: The Last Jedi	2017-12-15	Rian Johnson	$606,021,888	4	FALSE	TRUE
3	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	4	FALSE	TRUE
4	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	4	FALSE	TRUE
5	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	4	FALSE	TRUE
6	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	4	FALSE	TRUE
7	It	2017-09-08	Andy Muschietti	$327,481,748	4	FALSE	TRUE
8	Jumanji: Welcome to the Jungle	2017-12-20	Jake Kasdan	$320,537,066	2	TRUE	FALSE
9	Thor: Ragnarok	2017-11-03	Taika Waititi	$313,493,611	4	FALSE	TRUE
10	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	4	FALSE	TRUE
11	Justice League	2017-11-17	Zack Snyder	$227,733,120	4	FALSE	TRUE

(10) Comparing Dates

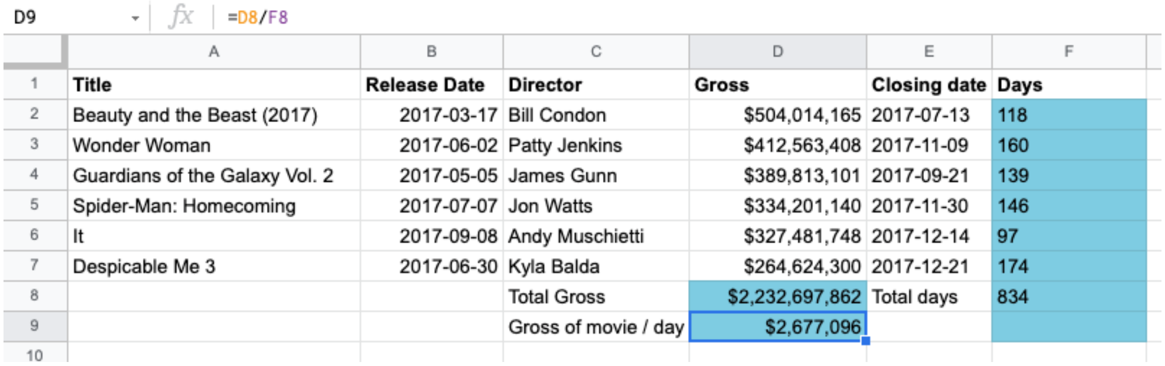
DATEDIF(start_date, end_date, unit): calculates the time difference between two dates. The difference will be calculated between start_date and end_date. The end_date must take place after the start_date. A third argument here is the unit, this can be:
"Y": the number of years between two dates
"M": the number of months between two dates
"D": the number of days between two dates
A full list can be found here
NOW(): a function without arguments, evaluates to the current time
For example, =DATEDIF("2018-01-01", "2018-01-03", "D") would evaluate to 2.



F3 =DATEDIF(B3,NOW(),"M")

	A	B	C	D	E	F
1	Title	Release Date	Director	Gross	Days ago	Months ago
2	Star Wars: The Last Jedi	2017-12-15	Rian Johnson	$606,021,888	1220	40
3	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	1493	49
4	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	1416	46
5	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	1444	47
6	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	1381	45
7	It	2017-09-08	Andy Muschietti	$327,481,748	1318	43
8	Jumanji: Welcome to the Jungle	2017-12-20	Jake Kasdan	$320,537,066	1215	39
9	Thor: Ragnarok	2017-11-03	Taika Waititi	$313,493,611	1262	41
10	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	1388	45
11	Justice League	2017-11-17	Zack Snyder	$227,733,120	1248	41

(11) Combining Functions



D9 =D8/F8

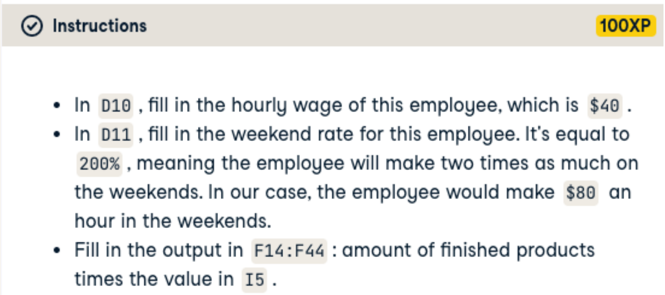
	A	B	C	D	E	F
1	Title	Release Date	Director	Gross	Closing date	Days
2	Beauty and the Beast (2017)	2017-03-17	Bill Condon	$504,014,165	2017-07-13	118
3	Wonder Woman	2017-06-02	Patty Jenkins	$412,563,408	2017-11-09	160
4	Guardians of the Galaxy Vol. 2	2017-05-05	James Gunn	$389,813,101	2017-09-21	139
5	Spider-Man: Homecoming	2017-07-07	Jon Watts	$334,201,140	2017-11-30	146
6	It	2017-09-08	Andy Muschietti	$327,481,748	2017-12-14	97
7	Despicable Me 3	2017-06-30	Kyla Balda	$264,624,300	2017-12-21	174
8			Total Gross	$2,232,697,862	Total days	834
9			Gross of movie / day	$2,677,096		
10						

2. Conditional Functions and Lookups

(1) Performance Statistics

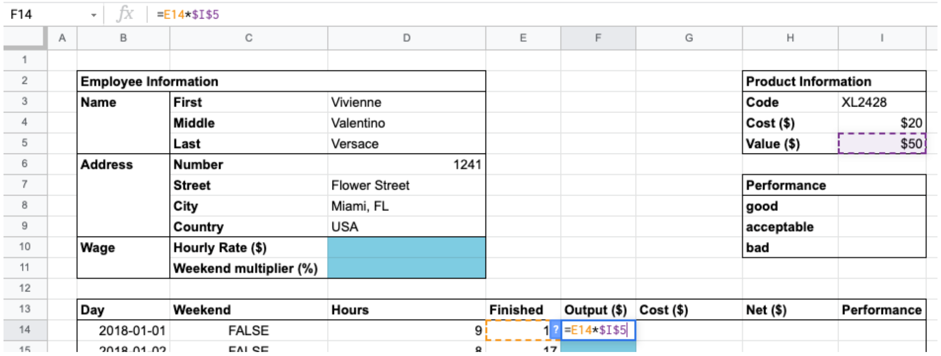
You work in a fashion company with 100 employees. You want to start tracking the effectiveness of your tailors and decide to keep track of their performance for the month January of 2018.

Finally, there's a bigger table, which contains the performance metrics:
Finished: the amount of finished products that day
Output: the combined value of those finished products
Cost: the cost to produce those products
Net: the difference between output and cost
Performance: the performance of the employee, bad, acceptable or good.



Instructions 100XP

- In `D10` , fill in the hourly wage of this employee, which is `$40` .
- In `D11` , fill in the weekend rate for this employee. It's equal to `200%` , meaning the employee will make two times as much on the weekends. In our case, the employee would make `$80` an hour in the weekends.
- Fill in the output in `F14:F44` : amount of finished products times the value in `I5` .

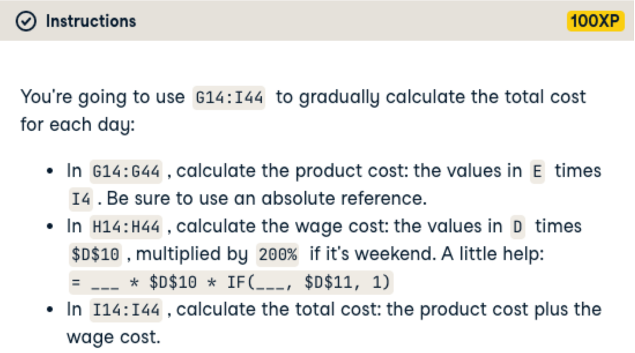


F14 fx =E14*$I$5

	A	B	C	D	E	F	G	H	I
1									
2		**Employee Information**						**Product Information**	
3		**Name**	**First**	Vivienne				**Code**	XL2428
4			**Middle**	Valentino				**Cost ($)**	$20
5			**Last**	Versace				**Value ($)**	$50
6		**Address**	**Number**	1241					
7			**Street**	Flower Street				**Performance**	
8			**City**	Miami, FL				**good**	
9			**Country**	USA				**acceptable**	
10		**Wage**	**Hourly Rate ($)**					**bad**	
11			**Weekend multiplier (%)**						
12									
13		**Day**	**Weekend**	**Hours**	**Finished**	**Output ($)**	**Cost ($)**	**Net ($)**	**Performance**
14		2018-01-01	FALSE	9	1	=E14*$I$5			
15		2018-01-02	FALSE	8	17				

(2) Flow Control- IF

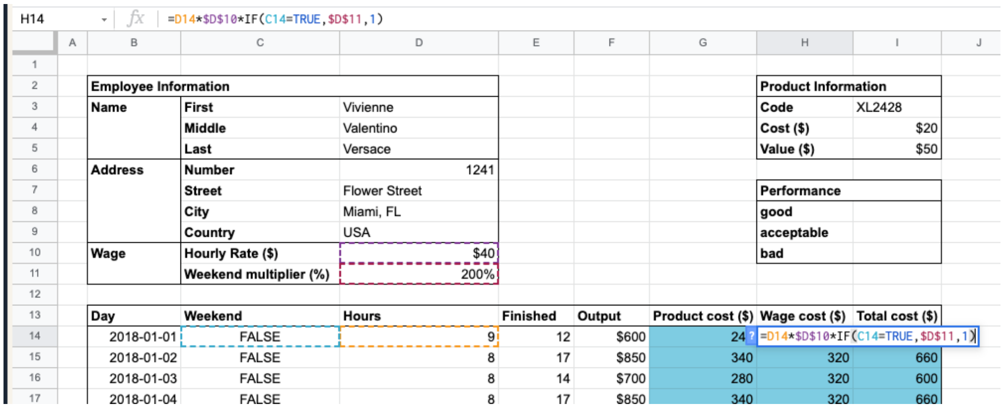
IF(logical_expression, value_if_true, value_if_false): depending on the logical_expression, return value_if_true when its result is TRUE, return value_if_false otherwise.



Instructions 100XP

You're going to use `G14:I44` to gradually calculate the total cost for each day:

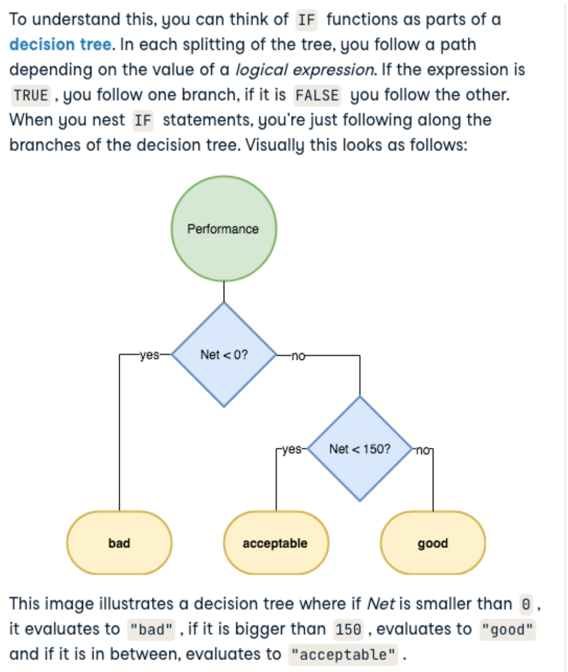
- In `G14:G44` , calculate the product cost: the values in `E` times `I4` . Be sure to use an absolute reference.
- In `H14:H44` , calculate the wage cost: the values in `D` times `$D$10` , multiplied by `200%` if it's weekend. A little help: `= ___ * $D$10 * IF(___, $D$11, 1)`
- In `I14:I44` , calculate the total cost: the product cost plus the wage cost.



H14 fx =D14*$D$10*IF(C14=TRUE,$D$11,1)

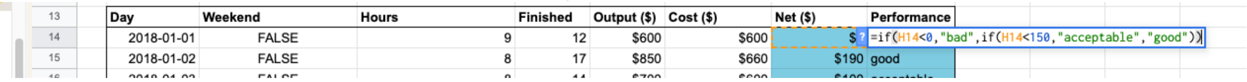
	A	B	C	D	E	F	G	H	I	J
1										
2		**Employee Information**						**Product Information**		
3		**Name**	**First**	Vivienne				**Code**	XL2428	
4			**Middle**	Valentino				**Cost ($)**	$20	
5			**Last**	Versace				**Value ($)**	$50	
6		**Address**	**Number**	1241						
7			**Street**	Flower Street				**Performance**		
8			**City**	Miami, FL				**good**		
9			**Country**	USA				**acceptable**		
10		**Wage**	**Hourly Rate ($)**	$40				**bad**		
11			**Weekend multiplier (%)**	200%						
12										
13		**Day**	**Weekend**	**Hours**	**Finished**	**Output**	**Product cost ($)**	**Wage cost ($)**	**Total cost ($)**	
14		2018-01-01	FALSE	9	12	$600	24	=D14*$D$10*IF(C14=TRUE,$D$11,1)		
15		2018-01-02	FALSE	8	17	$850	340	320	660	
16		2018-01-03	FALSE	8	14	$700	280	320	600	
17		2018-01-04	FALSE	8	17	$850	340	320	660	

(3) Nested Logical Functions – IF



To understand this, you can think of `IF` functions as parts of a **decision tree**. In each splitting of the tree, you follow a path depending on the value of a *logical expression*. If the expression is `TRUE`, you follow one branch, if it is `FALSE` you follow the other. When you nest `IF` statements, you're just following along the branches of the decision tree. Visually this looks as follows:

This image illustrates a decision tree where if *Net* is smaller than `0`, it evaluates to `"bad"`, if it is bigger than `150`, evaluates to `"good"` and if it is in between, evaluates to `"acceptable"`.



13	Day	Weekend	Hours	Finished	Output ($)	Cost ($)	Net ($)	Performance
14	2018-01-01	FALSE	9	12	$600	$600	$	=if(H14<0,"bad",if(H14<150,"acceptable","good"))
15	2018-01-02	FALSE	8	17	$850	$660	$190	good

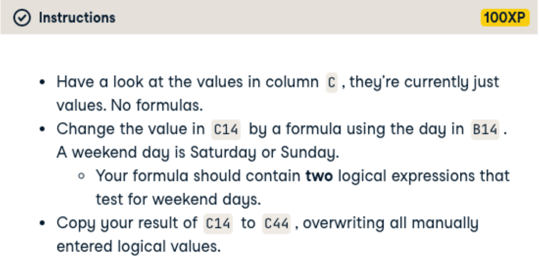
(4) Combining Logical values – OR, WEEKDAY

OR(logical_expression1, [logical_expression2, ...]): this is the logical operator that returns TRUE if one of the expressions is TRUE and FALSE if and only if all of them are FALSE.

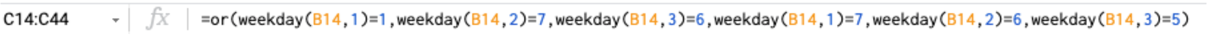
For example, we can determine whether a cell (e.g. A2) is equal to 21 or 22 by using the following formula: =OR(A2 = 21, A2 = 22).

WEEKDAY(date, [type]): evaluates to the day of the week of a date. type is 1, 2 or 3.

type = 1: Sunday is day 1 and Saturday is day 7 (default)
type = 2: Monday is day 1 and Sunday is day 7
type = 3: Monday is day 0 and Sunday is day 6



- Have a look at the values in column `C`, they're currently just values. No formulas.
- Change the value in `C14` by a formula using the day in `B14`. A weekend day is Saturday or Sunday.
  - Your formula should contain **two** logical expressions that test for weekend days.
- Copy your result of `C14` to `C44`, overwriting all manually entered logical values.



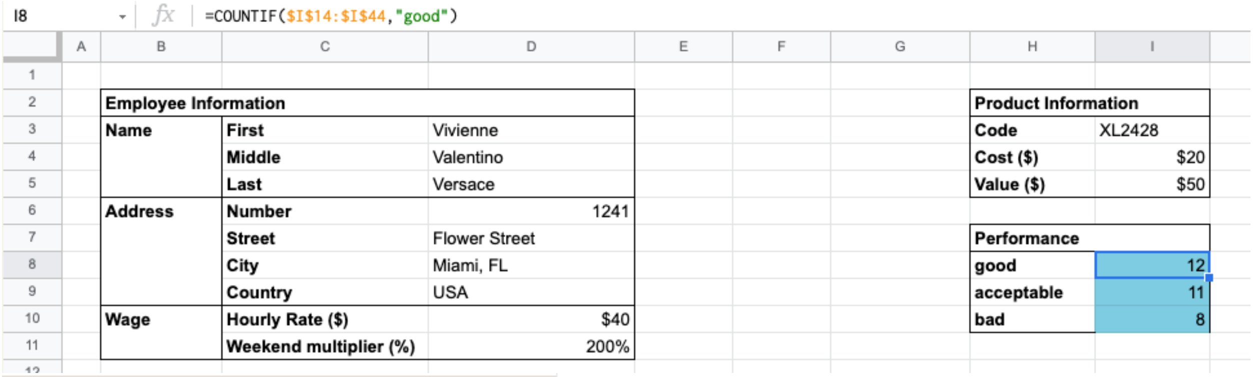
```
C14:C44   fx   =or(weekday(B14,1)=1,weekday(B14,2)=7,weekday(B14,3)=6,weekday(B14,1)=7,weekday(B14,2)=6,weekday(B14,3)=5)
```

(5) Conditional Counting – COUNTIF

COUNTIF(range, criterion): count the number of times the criterion is met in the specified range.

range: the source data that is used. Typically, you'll need to use an absolute reference for this one.

criterion: a pattern to check for. It can be as simple as a string you want to match on. For example: "good".

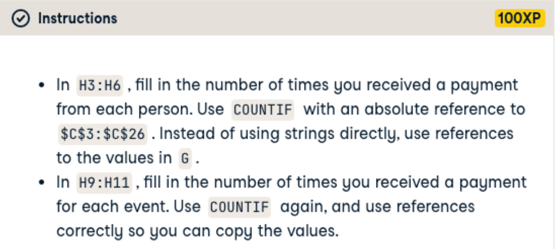


I8 =COUNTIF($I$14:$I$44,"good")

Employee Information		
Name	First	Vivienne
	Middle	Valentino
	Last	Versace
Address	Number	1241
	Street	Flower Street
	City	Miami, FL
	Country	USA
Wage	Hourly Rate ($)	$40
	Weekend multiplier (%)	200%

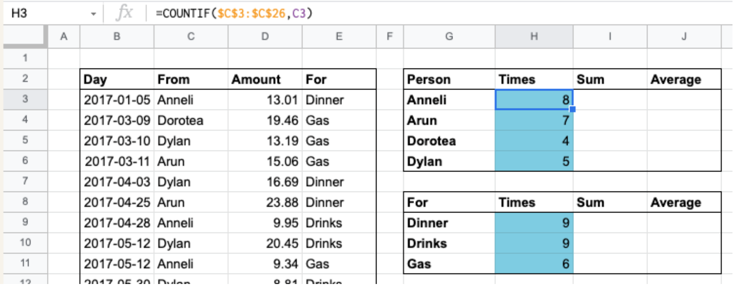
Product Information	
Code	XL2428
Cost ($)	$20
Value ($)	$50

Performance	
good	12
acceptable	11
bad	8



Instructions 100XP

- In H3:H6 , fill in the number of times you received a payment from each person. Use COUNTIF with an absolute reference to $C$3:$C$26 . Instead of using strings directly, use references to the values in G .
- In H9:H11 , fill in the number of times you received a payment for each event. Use COUNTIF again, and use references correctly so you can copy the values.



H3 =COUNTIF($C$3:$C$26,C3)

Day	From	Amount	For
2017-01-05	Anneli	13.01	Dinner
2017-03-09	Dorotea	19.46	Gas
2017-03-10	Dylan	13.19	Gas
2017-03-11	Arun	15.06	Gas
2017-04-03	Dylan	16.69	Dinner
2017-04-25	Arun	23.88	Dinner
2017-04-28	Anneli	9.95	Drinks
2017-05-12	Dylan	20.45	Drinks
2017-05-12	Anneli	9.34	Gas
2017-05-30	Dylan	8.81	Drinks

Person	Times	Sum	Average
Anneli	8		
Arun	7		
Dorotea	4		
Dylan	5		

For	Times	Sum	Average
Dinner	9		
Drinks	9		
Gas	6		

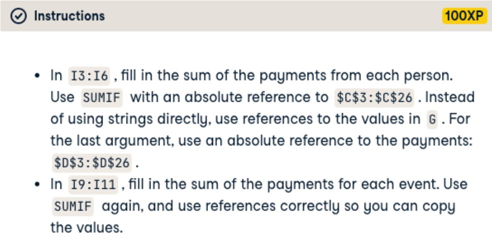
(6) Conditional Sum – SUMIF

SUMIF(range, criterion, sum_range): evaluates to the conditional sum across a range.

range: the range on which the criterion will be checked

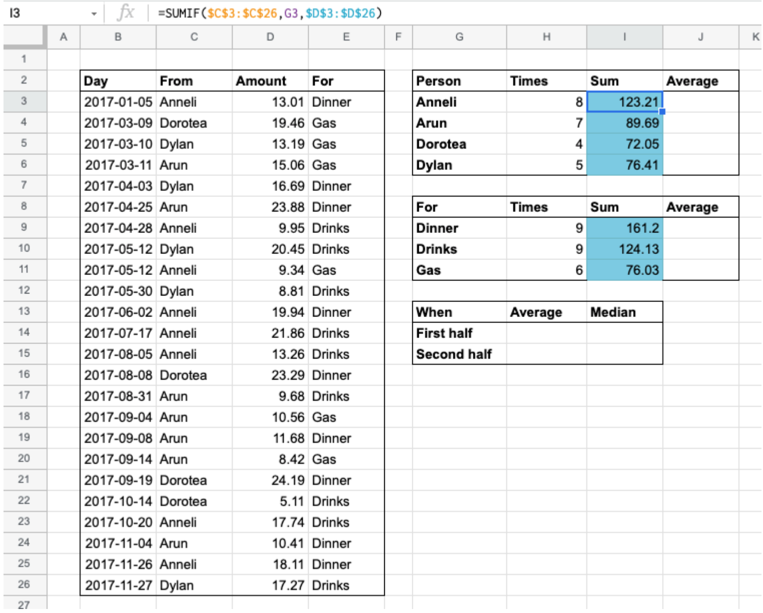
criterion: the pattern that will be checked, e.g. "Dylan"

sum_range: the range of values that will be summed up



Instructions 100XP

- In `I3:I6`, fill in the sum of the payments from each person. Use `SUMIF` with an absolute reference to `$C$3:$C$26`. Instead of using strings directly, use references to the values in `G`. For the last argument, use an absolute reference to the payments: `$D$3:$D$26`.
- In `I9:I11`, fill in the sum of the payments for each event. Use `SUMIF` again, and use references correctly so you can copy the values.



```
I3    fx  =SUMIF($C$3:$C$26,G3,$D$3:$D$26)
```

	A	B	C	D	E	F	G	H	I	J	K
1											
2		Day	From	Amount	For		Person	Times	Sum	Average	
3		2017-01-05	Anneli	13.01	Dinner		Anneli	8	123.21		
4		2017-03-09	Dorotea	19.46	Gas		Arun	7	89.69		
5		2017-03-10	Dylan	13.19	Gas		Dorotea	4	72.05		
6		2017-03-11	Arun	15.06	Gas		Dylan	5	76.41		
7		2017-04-03	Dylan	16.69	Dinner						
8		2017-04-25	Arun	23.88	Dinner		For	Times	Sum	Average	
9		2017-04-28	Anneli	9.95	Drinks		Dinner	9	161.2		
10		2017-05-12	Dylan	20.45	Drinks		Drinks	9	124.13		
11		2017-05-12	Anneli	9.34	Gas		Gas	6	76.03		
12		2017-05-30	Dylan	8.81	Drinks						
13		2017-06-02	Anneli	19.94	Dinner		When	Average	Median		
14		2017-07-17	Anneli	21.86	Drinks		First half				
15		2017-08-05	Anneli	13.26	Drinks		Second half				
16		2017-08-08	Dorotea	23.29	Dinner						
17		2017-08-31	Arun	9.68	Drinks						
18		2017-09-04	Arun	10.56	Gas						
19		2017-09-08	Arun	11.68	Dinner						
20		2017-09-14	Arun	8.42	Gas						
21		2017-09-19	Dorotea	24.19	Dinner						
22		2017-10-14	Dorotea	5.11	Drinks						
23		2017-10-20	Anneli	17.74	Drinks						
24		2017-11-04	Arun	10.41	Dinner						
25		2017-11-26	Anneli	18.11	Dinner						
26		2017-11-27	Dylan	17.27	Drinks						
27											

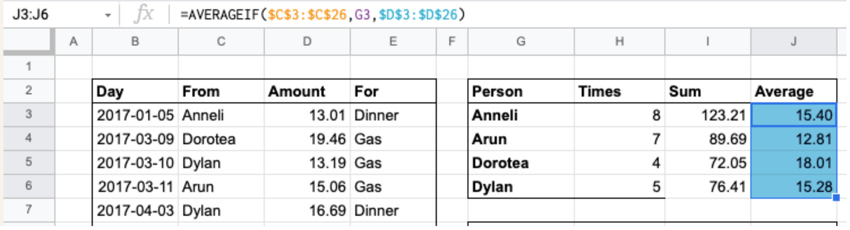
(7) Conditional Average – AVERAGEIF

AVERAGEIF(range, criterion, average_range): evaluates to the conditional average across a range.

range: the range on which the criterion will be checked

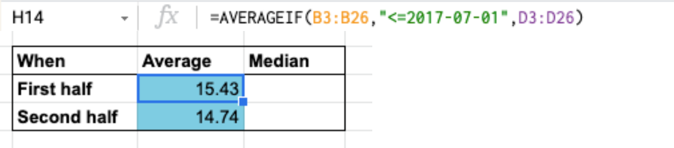
criterion: the pattern that will be checked, e.g. "Dylan"

average_range: the range of values that will be summed up



```
J3:J6    fx  =AVERAGEIF($C$3:$C$26,G3,$D$3:$D$26)
```

	A	B	C	D	E	F	G	H	I	J
1										
2		Day	From	Amount	For		Person	Times	Sum	Average
3		2017-01-05	Anneli	13.01	Dinner		Anneli	8	123.21	15.40
4		2017-03-09	Dorotea	19.46	Gas		Arun	7	89.69	12.81
5		2017-03-10	Dylan	13.19	Gas		Dorotea	4	72.05	18.01
6		2017-03-11	Arun	15.06	Gas		Dylan	5	76.41	15.28
7		2017-04-03	Dylan	16.69	Dinner					



```
H14    fx  =AVERAGEIF(B3:B26,"<=2017-07-01",D3:D26)
```

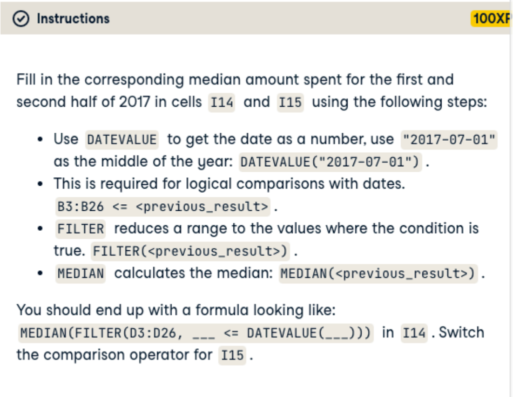
When	Average	Median
First half	15.43	
Second half	14.74	

(8) Filters – FILTER, DATEVALUE, MEDIAN

Finally, you'll have to find the conditional median on a range. However, there's no such function as MEDIANIF, so you'll have to find a way to generalize what you've learned previously.
You can do so using a filter. A filter will take a range, apply a condition to all values of it and evaluate to the range of values where the condition passed. Specifically, you'll be using the following:

FILTER(range, condition1, [condition2, ...]): evaluates to a filtered version of range, based on the passed conditions. condition1 here is substantially different from the criterion argument you're used to. condition1 is not a string, but rather a range of logical values, for example A1:A5 > 5.

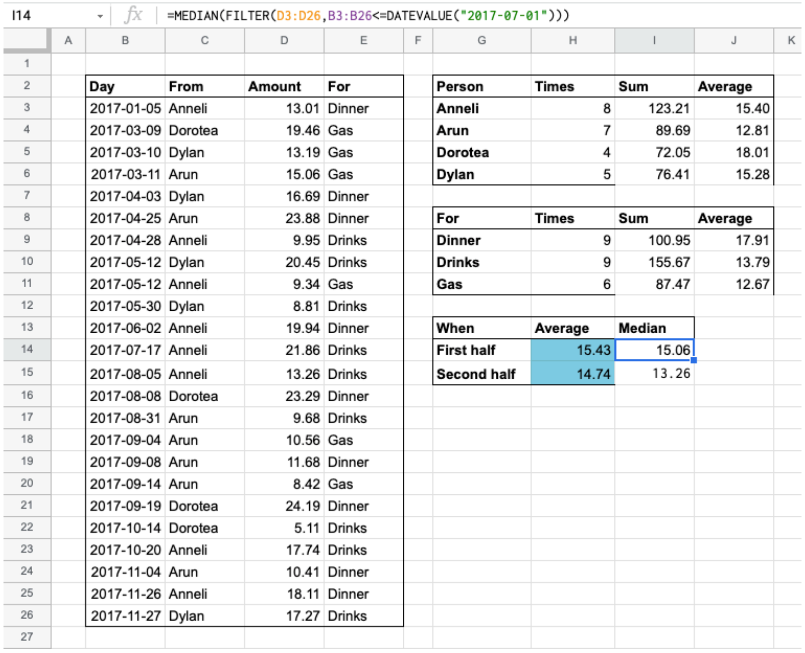
For example, if we wanted to calculate the average amount spent on dinners, we could use the following formula: =AVERAGE(FILTER(D3:D26, E3:E26 = "Dinner")). Here, we filter the range of amount spent (D3:D26) based on whether the range E3:E26 contains the word "Dinner". We then take the average of this filtered range.
DATEVALUE(date_string): evaluates to the date object of a date_string



Fill in the corresponding median amount spent for the first and second half of 2017 in cells `I14` and `I15` using the following steps:

- Use `DATEVALUE` to get the date as a number, use `"2017-07-01"` as the middle of the year: `DATEVALUE("2017-07-01")`.
- This is required for logical comparisons with dates. `B3:B26 <= <previous_result>`.
- `FILTER` reduces a range to the values where the condition is true. `FILTER(<previous_result>)`.
- `MEDIAN` calculates the median: `MEDIAN(<previous_result>)`.

You should end up with a formula looking like: `MEDIAN(FILTER(D3:D26, ___ <= DATEVALUE(___)))` in `I14`. Switch the comparison operator for `I15`.



I14 =MEDIAN(FILTER(D3:D26,B3:B26<=DATEVALUE("2017-07-01")))

Day	From	Amount	For
2017-01-05	Anneli	13.01	Dinner
2017-03-09	Dorotea	19.46	Gas
2017-03-10	Dylan	13.19	Gas
2017-03-11	Arun	15.06	Gas
2017-04-03	Dylan	16.69	Dinner
2017-04-25	Arun	23.88	Dinner
2017-04-28	Anneli	9.95	Drinks
2017-05-12	Dylan	20.45	Drinks
2017-05-12	Anneli	9.34	Gas
2017-05-30	Dylan	8.81	Drinks
2017-06-02	Anneli	19.94	Dinner
2017-07-17	Anneli	21.86	Drinks
2017-08-05	Anneli	13.26	Drinks
2017-08-08	Dorotea	23.29	Dinner
2017-08-31	Arun	9.68	Drinks
2017-09-04	Arun	10.56	Gas
2017-09-08	Arun	11.68	Dinner
2017-09-14	Arun	8.42	Gas
2017-09-19	Dorotea	24.19	Dinner
2017-10-14	Dorotea	5.11	Drinks
2017-10-20	Anneli	17.74	Drinks
2017-11-04	Arun	10.41	Dinner
2017-11-26	Anneli	18.11	Dinner
2017-11-27	Dylan	17.27	Drinks

Person	Times	Sum	Average
Anneli	8	123.21	15.40
Arun	7	89.69	12.81
Dorotea	4	72.05	18.01
Dylan	5	76.41	15.28

For	Times	Sum	Average
Dinner	9	100.95	17.91
Drinks	9	155.67	13.79
Gas	6	87.47	12.67

When	Average	Median
First half	15.43	15.06
Second half	14.74	13.26

(9) Grades in class

a. Automating the lookup – VLOOKUP

VLOOKUP(search_key, range, index, is_sorted): look for a match in the leftmost column of a lookup table and return the value in a certain column:

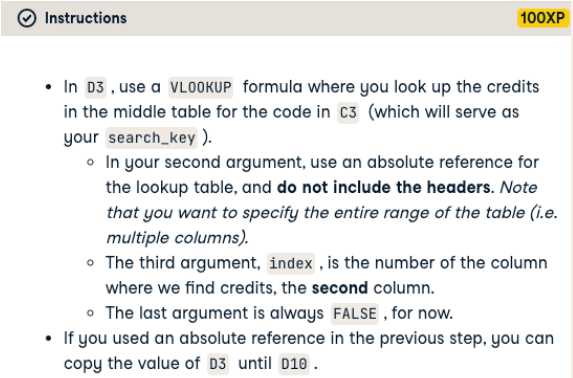
search_key: the value to search for

range: the lookup table, without the headers. You typically use an absolute reference for this.

index: the column number of the value to be returned, where the first column in range is numbered 1

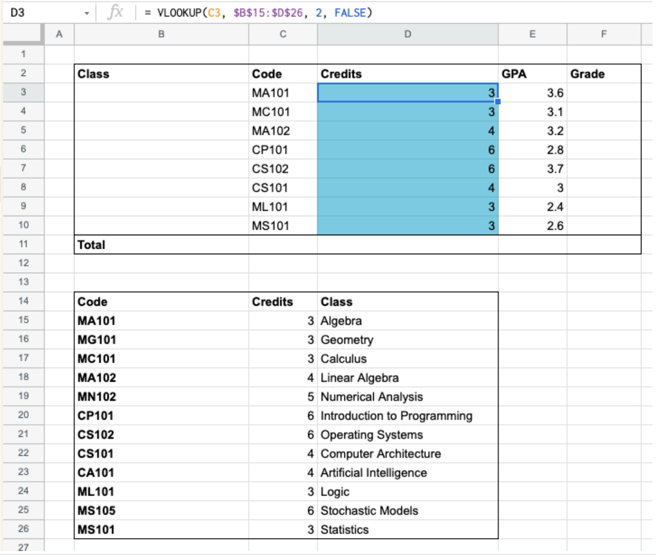
is_sorted: should be FALSE for now

You can compare it to the process of looking through a phone book. The search_key would be the name of the person you want the phone number of. The range is the data in the book, with the names in the leftmost column. Finally, the index is the number of the column where you find what you need, the phone number.



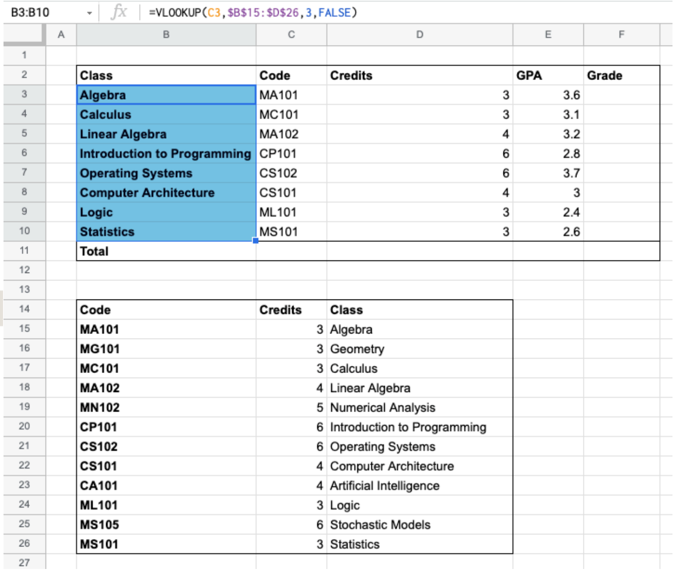
Instructions 100XP

- In `D3`, use a `VLOOKUP` formula where you look up the credits in the middle table for the code in `C3` (which will serve as your `search_key`).
  - In your second argument, use an absolute reference for the lookup table, and **do not include the headers**. *Note that you want to specify the entire range of the table (i.e. multiple columns).*
  - The third argument, `index`, is the number of the column where we find credits, the **second** column.
  - The last argument is always `FALSE`, for now.
- If you used an absolute reference in the previous step, you can copy the value of `D3` until `D10`.



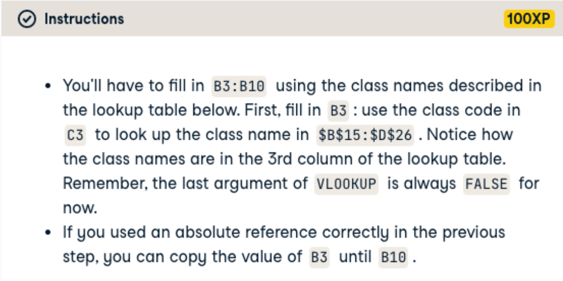
D3 = VLOOKUP(C3, $B$15:$D$26, 2, FALSE)

	A	B	C	D	E	F
1						
2		**Class**	**Code**	**Credits**	**GPA**	**Grade**
3			MA101	3	3.6	
4			MC101	3	3.1	
5			MA102	4	3.2	
6			CP101	6	2.8	
7			CS102	6	3.7	
8			CS101	4	3	
9			ML101	3	2.4	
10			MS101	3	2.6	
11		**Total**				
12						
13						
14		**Code**	**Credits**	**Class**		
15		**MA101**	3	Algebra		
16		**MG101**	3	Geometry		
17		**MC101**	3	Calculus		
18		**MA102**	4	Linear Algebra		
19		**MN102**	5	Numerical Analysis		
20		**CP101**	6	Introduction to Programming		
21		**CS102**	6	Operating Systems		
22		**CS101**	4	Computer Architecture		
23		**CA101**	4	Artificial Intelligence		
24		**ML101**	3	Logic		
25		**MS105**	6	Stochastic Models		
26		**MS101**	3	Statistics		
27						



B3:B10 =VLOOKUP(C3,$B$15:$D$26,3,FALSE)

	A	B	C	D	E	F
1						
2		**Class**	**Code**	**Credits**	**GPA**	**Grade**
3		**Algebra**	MA101	3	3.6	
4		**Calculus**	MC101	3	3.1	
5		**Linear Algebra**	MA102	4	3.2	
6		**Introduction to Programming**	CP101	6	2.8	
7		**Operating Systems**	CS102	6	3.7	
8		**Computer Architecture**	CS101	4	3	
9		**Logic**	ML101	3	2.4	
10		**Statistics**	MS101	3	2.6	
11		**Total**				
12						
13						
14		**Code**	**Credits**	**Class**		
15		**MA101**	3	Algebra		
16		**MG101**	3	Geometry		
17		**MC101**	3	Calculus		
18		**MA102**	4	Linear Algebra		
19		**MN102**	5	Numerical Analysis		
20		**CP101**	6	Introduction to Programming		
21		**CS102**	6	Operating Systems		
22		**CS101**	4	Computer Architecture		
23		**CA101**	4	Artificial Intelligence		
24		**ML101**	3	Logic		
25		**MS105**	6	Stochastic Models		
26		**MS101**	3	Statistics		
27						



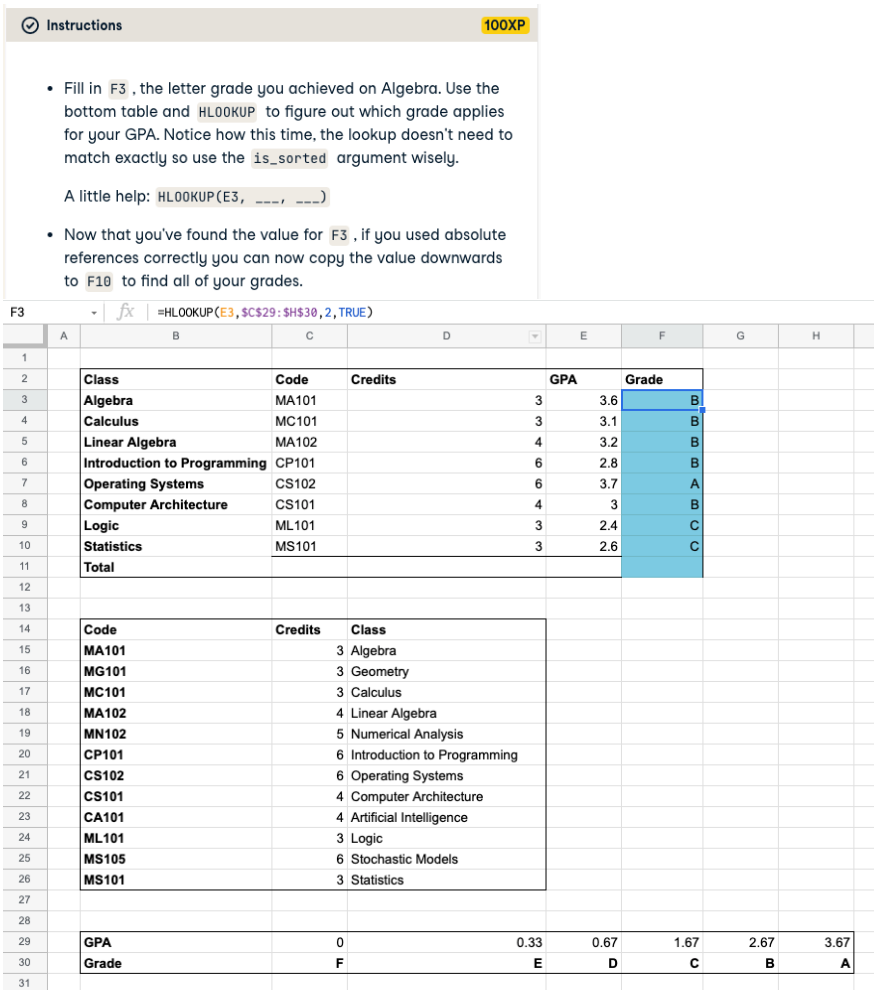
Instructions 100XP

- You'll have to fill in `B3:B10` using the class names described in the lookup table below. First, fill in `B3`: use the class code in `C3` to look up the class name in `$B$15:$D$26`. Notice how the class names are in the 3rd column of the lookup table. Remember, the last argument of `VLOOKUP` is always `FALSE` for now.
- If you used an absolute reference correctly in the previous step, you can copy the value of `B3` until `B10`.

b. Horizontal Lookup – HLOOKUP
HLOOKUP(search_key, range, index, is_sorted): similar to VLOOKUP but in a horizontal fashion. The key will be looked for in the uppermost row, and index now refers to the row number.

the last argument, is_sorted. If set to TRUE (default), the function assumes that the values in range are sorted. When this is the case, the match doesn't have to be exact, but HLOOKUP will look for the closest match less than or equal to search_key. If search_key is FALSE, an exact match is required.

For example, =HLOOKUP(0.57, $C$29:$H$30, 2, TRUE) would evaluate to E in the given spreadsheet, as the closest match less than or equal to 0.57 is 0.33.



- Fill in `F3`, the letter grade you achieved on Algebra. Use the bottom table and `HLOOKUP` to figure out which grade applies for your GPA. Notice how this time, the lookup doesn't need to match exactly so use the `is_sorted` argument wisely.

  A little help: `HLOOKUP(E3, ___, ___)`

- Now that you've found the value for `F3`, if you used absolute references correctly you can now copy the value downwards to `F10` to find all of your grades.

```
F3    fx  =HLOOKUP(E3,$C$29:$H$30,2,TRUE)
```

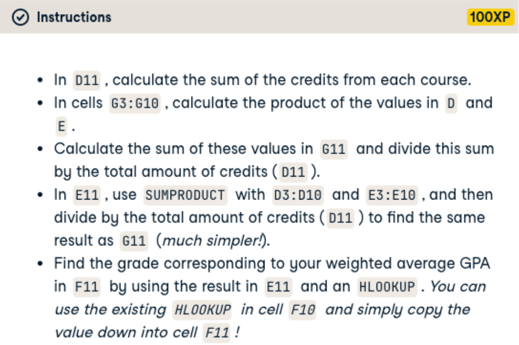
	A	B	C	D	E	F	G	H
1								
2		**Class**	**Code**	**Credits**	**GPA**	**Grade**		
3		**Algebra**	MA101	3	3.6	B		
4		**Calculus**	MC101	3	3.1	B		
5		**Linear Algebra**	MA102	4	3.2	B		
6		**Introduction to Programming**	CP101	6	2.8	B		
7		**Operating Systems**	CS102	6	3.7	A		
8		**Computer Architecture**	CS101	4	3	B		
9		**Logic**	ML101	3	2.4	C		
10		**Statistics**	MS101	3	2.6	C		
11		**Total**						
12								
13								
14		**Code**	**Credits**	**Class**				
15		**MA101**	3	Algebra				
16		**MG101**	3	Geometry				
17		**MC101**	3	Calculus				
18		**MA102**	4	Linear Algebra				
19		**MN102**	5	Numerical Analysis				
20		**CP101**	6	Introduction to Programming				
21		**CS102**	6	Operating Systems				
22		**CS101**	4	Computer Architecture				
23		**CA101**	4	Artificial Intelligence				
24		**ML101**	3	Logic				
25		**MS105**	6	Stochastic Models				
26		**MS101**	3	Statistics				
27								
28								
29		**GPA**	0	0.33	0.67	1.67	2.67	3.67
30		**Grade**	F	E	D	C	B	A
31								

c. Weighted Average – SUMPRODUCT, HLOOKUP<加权求和>

SUMPRODUCT(array1, [array2, ...]): figure out the sum of products of 2 or more ranges of equal size.

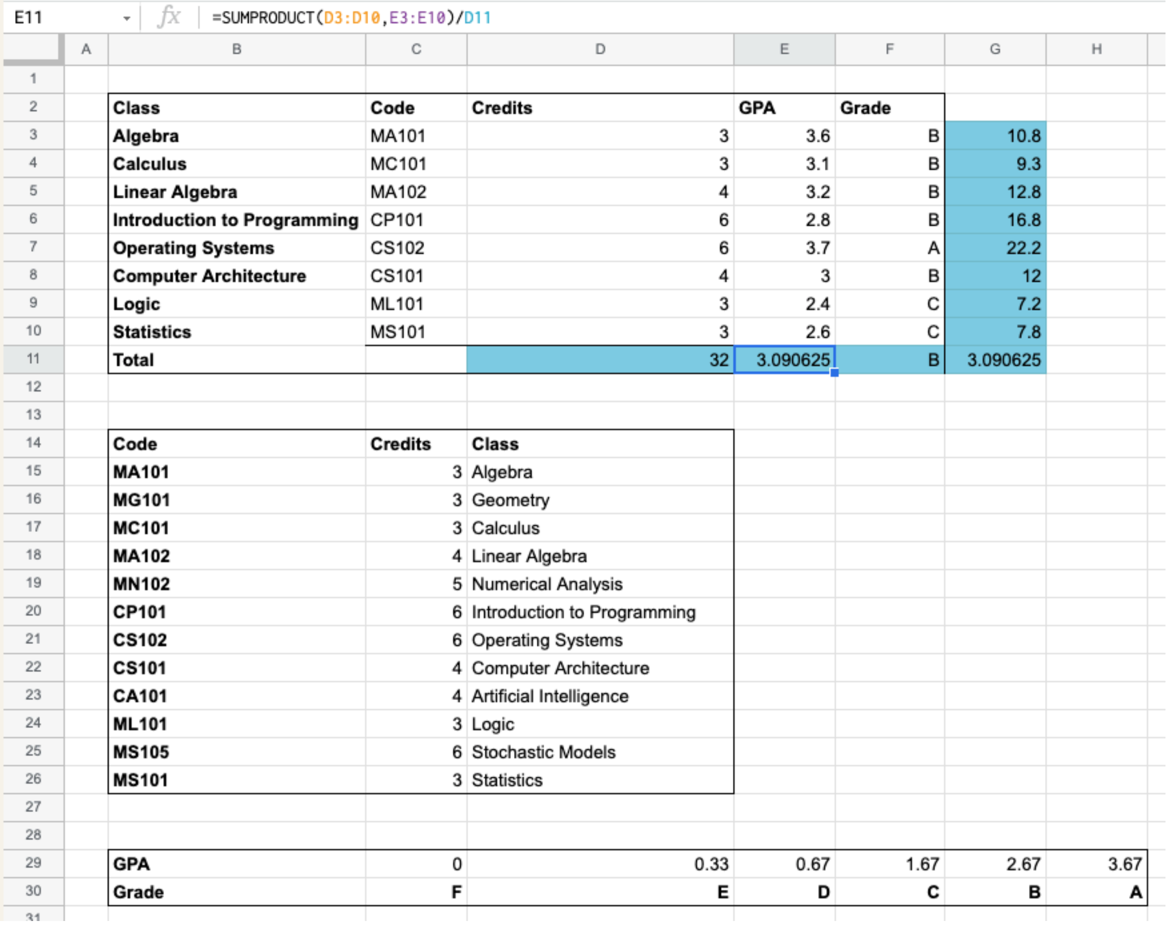
E.g. SUMPRODUCT(A1:A3, B1:B3) evaluates to the result of (A1 * B1) + (A2 * B2)+ (A3 * B3). In mathematics, this operation is called the dot product.

In addition, you will again need to use HLOOKUP to calculate your grade: HLOOKUP(search_key, range, index, is_sorted)



Instructions 100XP

- In `D11`, calculate the sum of the credits from each course.
- In cells `G3:G10`, calculate the product of the values in `D` and `E`.
- Calculate the sum of these values in `G11` and divide this sum by the total amount of credits (`D11`).
- In `E11`, use `SUMPRODUCT` with `D3:D10` and `E3:E10`, and then divide by the total amount of credits (`D11`) to find the same result as `G11` (*much simpler!*).
- Find the grade corresponding to your weighted average GPA in `F11` by using the result in `E11` and an `HLOOKUP`. *You can use the existing `HLOOKUP` in cell `F10` and simply copy the value down into cell `F11`!*



E11 | fx | =SUMPRODUCT(D3:D10,E3:E10)/D11

	A	B	C	D	E	F	G	H
1								
2		**Class**	**Code**	**Credits**	**GPA**	**Grade**		
3		**Algebra**	MA101	3	3.6	B	10.8	
4		**Calculus**	MC101	3	3.1	B	9.3	
5		**Linear Algebra**	MA102	4	3.2	B	12.8	
6		**Introduction to Programming**	CP101	6	2.8	B	16.8	
7		**Operating Systems**	CS102	6	3.7	A	22.2	
8		**Computer Architecture**	CS101	4	3	B	12	
9		**Logic**	ML101	3	2.4	C	7.2	
10		**Statistics**	MS101	3	2.6	C	7.8	
11		**Total**		32	3.090625	B	3.090625	
12								
13								
14		**Code**	**Credits**	**Class**				
15		**MA101**	3	Algebra				
16		**MG101**	3	Geometry				
17		**MC101**	3	Calculus				
18		**MA102**	4	Linear Algebra				
19		**MN102**	5	Numerical Analysis				
20		**CP101**	6	Introduction to Programming				
21		**CS102**	6	Operating Systems				
22		**CS101**	4	Computer Architecture				
23		**CA101**	4	Artificial Intelligence				
24		**ML101**	3	Logic				
25		**MS105**	6	Stochastic Models				
26		**MS101**	3	Statistics				
27								
28								
29		**GPA**	0	0.33	0.67	1.67	2.67	3.67
30		**Grade**	**F**	**E**	**D**	**C**	**B**	**A**

Reference:

Google Sheets Function List:
https://support.google.com/docs/table/25273?visit_id=637543556758785226-3467028831&rd=2

Hard Coded:
https://en.wikipedia.org/wiki/Hard_coding