#### Conditional functions and lookups

In the last chapter of the course, you'll master more advanced functions like IF() and VLOOKUP(). Conditional and lookup functions won’t seem so scary after you completed this chapter.

#### 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.

To the right, you can see an example sheet of the performance metrics for a tailor called Vivienne V. Versace. There are some tables with the employee and product information and a performance table.

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.

Let's get to work.

##### Instructions

* 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.

#### Flow control - IF

Throughout the previous chapters, there have been several occasions where you had to use **logical** values: TRUE or FALSE. Now here's where they get really useful: **flow control functions**.

These are functions that use a certain logical value as one of their arguments and evaluate according to that value. One will be specifically important in this exercise:

* [**IF(logical\_expression, value\_if\_true, value\_if\_false)**](https://support.google.com/docs/answer/3093364): depending on the logical\_expression, return value\_if\_true when its result is TRUE, return value\_if\_false otherwise.

For example, = IF(TRUE, 2, -2) would evaluate to 2.

##### Instructions

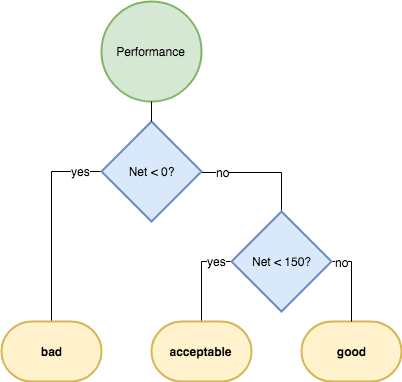
You're going to use G14:I44 to gradually calculate the total cost for each day:

* 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.

#### Nested logical functions - IF

We've merged the cost you calculated in the previous exercise into one column, G, to win some space. [Biraz yer kazanmak için önceki alıştırmada hesapladığınız maliyeti tek bir sütun olan G'de birleştirdik.] Now, it's time to dig deeper, you're going to use a nested if statement. [Şimdi, daha derine inmenin zamanı geldi, iç içe bir if ifadesi kullanacaksınız.]

To understand this, you can think of IF functions as parts of a [**decision tree**](https://en.wikipedia.org/wiki/Decision_tree). [Bunu anlamak için EĞER fonksiyonlarını bir karar ağacının parçaları olarak düşünebilirsiniz.] In each splitting of the tree, you follow a path depending on the value of a logical expression. [Ağacın her bölünmesinde mantıksal bir ifadenin değerine bağlı olarak bir yol izlersiniz.] If the expression is TRUE, you follow one branch, if it is FALSE you follow the other. [İfade DOĞRU ise bir dalı, YANLIŞ ise diğerini takip edersiniz.] When you nest IF statements, you're just following along the branches of the decision tree. [EĞER deyimlerini yuvaladığınızda, karar ağacının dallarını takip ediyorsunuz.] Visually this looks as follows: [Görsel olarak bu şöyle görünür:]



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". [Bu görüntü, Net 0'dan küçükse 'kötü', 150'den büyükse 'iyi' ve aradaysa 'kabul edilebilir' olarak değerlendirdiği bir karar ağacını göstermektedir.]

##### Instructions

* Fill in the net income in H14:H44: the output minus the cost.
* Use the logic that is shown in the image above to fill in I14:I44. In words: if the net cost is less than $0, performance should be "bad". Otherwise if the net cost is less than $150 the performance should be "acceptable". Otherwise the performance should be "good".

A little help: IF(H14 < \_\_\_, \_\_\_, IF(H14 < \_\_\_, "acceptable", \_\_\_)).

#### Combining logical values - OR, WEEKDAY

It's time to revise a column you previously left untouched: weekend. The value is either TRUE or FALSE. Right now, all the values are filled in manually, it's [**hard coded**](https://en.wikipedia.org/wiki/Hard_coding). We can do better using the following formulas:

* [**OR(logical\_expression1, [logical\_expression2, ...])**](https://support.google.com/docs/answer/3093306): 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])**](https://support.google.com/docs/answer/3092985): 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

To help get you started, a logical expression to test whether a date in cell *B2* fell on a Monday could look like this: WEEKDAY(B2) = 2

##### Instructions

* 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.

#### Conditional counting - COUNTIF

Finally, you'll need to fill in the frequency table in I8:I10. You can do this by using the following function in Google Sheets:

* [**COUNTIF(range, criterion)**](https://support.google.com/docs/answer/3093480): 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". You'll see more complex criterions in later exercises.

For example: if A1:A3 contains "good", "bad", "bad", then = COUNTIF(A1:A3, "bad") evaluates to 2.

##### Instructions

* In I8, fill in the number of times the word "good" appears in I14:I44. Don't use "good" directly, but rather use a reference to H8. Make sure to use an absolute reference in the first argument.

A little help: \_\_\_($I$14:$I$44, \_\_\_).

* If you used an absolute reference in the previous instruction, you can simply copy the value in I8 to I10 to find the frequencies of "acceptable" and "bad".

#### Conditional aggregation - COUNTIF

Let's dive a bit deeper into the world of **conditional aggregation** functions. These functions can be used to calculate summary statistics for each category of data, like COUNTIF.

The data you'll be working with is a set of payments for dinner, gas, and drinks that four friends made to you in 2017.

A refresher:

* [**COUNTIF(range, criterion)**](https://support.google.com/docs/answer/3093480): 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, e.g. "Dylan". You'll see more complex criterions in later exercises.

For example if A1:A3 holds "Arun", "Dylan", "Dylan", then = COUNTIF(A1:A3, "Dylan") evaluates to 2.

##### Instructions

* 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.

#### Conditional sum - SUMIF

Things get a bit more complex when the range to check the criterion on is not the same as the range we want the statistics for.

This can be the case for SUMIF:

* [**SUMIF(range, criterion, sum\_range)**](https://support.google.com/docs/answer/3093583): 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

For example if A1:A3 holds "Arun", "Dylan", "Dylan" and B1:B3 has 3, 4, 8, then = SUMIF(A1:A3, "Dylan", B1:B3) evaluates to 12.

##### Instructions

* 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.

#### Conditional average - AVERAGEIF

Another interesting statistic you can calculate grouped per all the categories is the average.

You're going to be using AVERAGEIF:

* [**AVERAGEIF(range, criterion, average\_range)**](https://support.google.com/docs/answer/3256529): 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

For example if A1:A3 holds "Arun", "Dylan", "Dylan" and B1:B3 has 3, 4, 8, then = AVERAGEIF(A1:A3, "Dylan", B1:B3) evaluates to 6.

##### Instructions

* In J3:J6, fill in the average of the payments from each person. Use AVERAGEIF 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 J9:J11, fill in the average of the payments for each event. Use AVERAGEIF again, and use references correctly so you can copy the values.

#### Advanced conditions - AVERAGEIF

Up until now, the condition was always an equality check on certain ranges. However, as specified in [**the documentation**](https://support.google.com/docs/answer/3256529), the criterion argument can be a bit more advanced:

* Equals: 1 or "= 1"
* Greater than: "> 1"
* Greater than or equal to: ">= 1"
* Less than: "< 1"
* Less than or equal to: "<= 1"
* Not equal to: "<> 1"

Note that this is very similar to the comparison operators you saw earlier.

As a refresher, here's the signature of the AVERAGEIF function:

* [**AVERAGEIF(range, criterion, average\_range)**](https://support.google.com/docs/answer/3256529): 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

##### Instructions

* In H14, fill in the average amount of the payments you received in the first half of the year. You can say that this the period before or on the first of July, 2017 -- or "<= 2017-07-01". Make sure you pass the correct range for the condition.
* In H15, fill in the average amount of the payments you received in the second half of the year, or strictly after the first of July, 2017.

#### 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, ...])**](https://support.google.com/docs/answer/3093197): 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)**](https://support.google.com/docs/answer/3093039): evaluates to the date object of a date\_string

##### Instructions

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.

#### Grades in class

In the last group of exercises in this course, you'll be learning about the concept of **lookup** in Google Sheets. Lookups work similarly to how you'd look up phone numbers in a phonebook.

You'll see some exact definitions later, first let's have a look at the data. You'll be working with some grades that a university student achieved on her courses. She asked you to do some analysis. She gave you three tables:

* The actual grades are at the top. These are given in [**GPA**](https://en.wikipedia.org/wiki/Grading_(education)). There are some empty columns which you'll fill in throughout the exercises.
* In the middle, there's a table with some specifics on the courses.
* On the bottom, there's a lookup table to convert GPA to a letter grade (cf. [**Wikipedia**](https://en.wikipedia.org/wiki/Grading_systems_by_country#United_States))

##### Instructions

* In D3, find the row in the middle table associated with the course with code MA101. Fill in the credits you find for that course. No need to use formulas now, just manual work.
* Repeat the process for D4:D6. Without using a formula, this is a lot of manual work indeed so we don't want you to fill the whole table.

#### Automating the lookup - VLOOKUP

Introducing VLOOKUP:

* [**VLOOKUP(search\_key, range, index, is\_sorted)**](https://support.google.com/docs/answer/3093318): 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

* 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.

#### More about lookup - VLOOKUP

Let's talk a bit more about VLOOKUP:

* [**VLOOKUP(search\_key, range, index, is\_sorted)**](https://support.google.com/docs/answer/3093318)

The index argument is probably the most difficult to grasp. The confusing part here is that Google Sheets normally uses letters to define columns. However, index actually refers to the number of the column, relative to the range you defined. E.g. the leftmost column in the range would be defined with index 1, the third column would have index 3.

An example, A has: "ML101", "CP101" and B has: 3, 6. Then =VLOOKUP("CP101", $A$1:$B$2, 2, FALSE) evaluates to 6.

##### Instructions

* 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.

#### Horizontal lookup - HLOOKUP

Although way less common, it can be useful to do a lookup through horizontally organized data. Introducing HLOOKUP:

* [**HLOOKUP(search\_key, range, index, is\_sorted)**](https://support.google.com/docs/answer/3093375): 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.

You're now going to use 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.

##### Instructions

* 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.

#### Weighted average - SUMPRODUCT, HLOOKUP

In this last exercise, you'll calculate an average GPA and grade. To do so, the following function might come in handy:

* [**SUMPRODUCT(array1, [array2, ...])**](https://support.google.com/docs/answer/3094294): 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**](https://en.wikipedia.org/wiki/Dot_product).

In addition, you will again need to use HLOOKUP to calculate your grade:

* [**HLOOKUP(search\_key, range, index, is\_sorted)**](https://support.google.com/docs/answer/3093375)

##### Instructions

* 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*!