# PROGRAMMING LANGUAGES

**By:** Asier Larrazabal, David Fernández and Lingfeng Chen
**Github:** Repositorio

# Datasets

## Popularity

Source:
https://www.kaggle.com/datasets/muhammadkhalid/most-popular-programming-languages-since-2004
Description:
This dataset contains the percentage of each language from 1 to 100%  based con github repos provided by github

## Issues

Source:
https://www.kaggle.com/datasets/muhammadkhalid/most-popular-programming-languages-since-2004
Description:
This dataset contains all issues that have been reported in github repos separated for each quarter of year.

## StackOverflow

Source:
https://insights.stackoverflow.com/survey
Description:
This is a huge dataset that contains a wide variety of information based on answers of users of Stack Overflow

In order to reduce the huge size of the dataset we had to eliminate rows and columns with a python script using Pandas.
(80000 rows,150MB) -> (20000 rows, 9,7MB)

# Popularity of each programming language

## Data collection (Popularity)

This dataset contains the percentage of use of each programing languages divided by the month of each, the data obtained is between July of 2004 and May of 2023

To read the file, we use the function read.csv, so we have to take in account that it has a header, the data was separated by “,”, the decimal numbers were written with “.” and the Na values were found as “NA”.

## Data preparation

To work better with the data frame, we create another dataframe with the values we are going to work with, so it is easier.

## Data exploration and Model Deployment

In shiny, we create 2 slider inputs, one for the month and another one for the year. With these 2 values we call the function to create the pie, and we create it next to the inputs. Also we put a title before this pie, and a subgraph under the plot to explain its limits.
The less important data, we combine it into one simple value, named as “Others”.

# Raising languages

## Data collection (Issues)

This dataset contains all the issues reported in github related to each language, which can let us know which languages are currently used in each repos/project. Languages such as C were popular at the beginning of computer science but they may not be popular today, then using this data we can know which languages are emerging.

In order to obtain the line plot I had to sort the dataset by Language > Date and create another statistic which is the cumulative frequency of issues of each language. Besides, all the NA values or missing date data were considered as 0.

To read the csv file, I used the read.csv file considering that it have header, “,” as separator, “.” for decimal numbers (Even we don't have decimal numbers), “NA” as NA value, and fill missing data with NA

## Data preparation

I created a value for the cumulative frequency of the issues because in that way users can observe a rising curve that reflects how each language's growth in Github and compare them in the shiny App. I considered NA values as 0 because if we do not have the data the curve does not grow.

Maybe I could have integrated the number of percentages of each language in the repositories of github, or the number of questions in stack overflow with each language..

## Data exploration and Model Deployment

The shiny app allows you to compare 2 different languages. It would be possible to add a huge graph with all the curves that contains all the lines but it would create more chaos in the plot and take a long time to load everything so I considered this as the best way.

# Most want to work with languages

## Data collection (StackOverflow)

This dataset contains a column named “LanguageWantToWorkWith”, that have all the programing languages that are required by a lot of programers

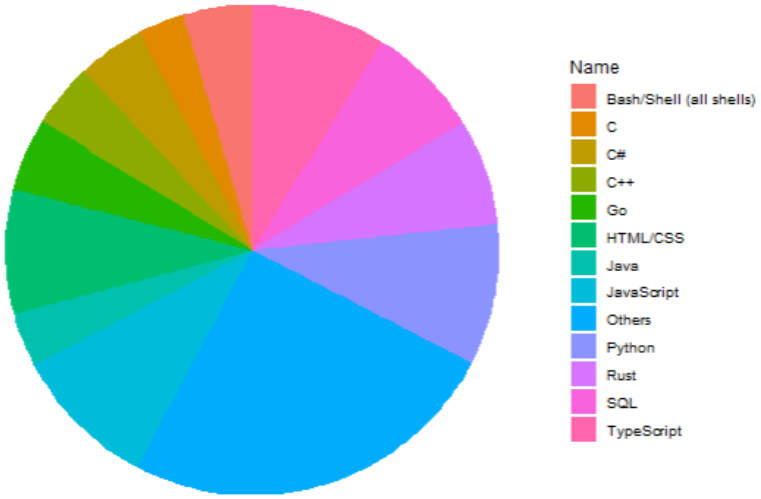
To read the file, we use the function read.csv, so we have to take in account that it has a header, the data was separated by “,”, the decimal numbers were written with “.” and the Na values were found as “NA”.

## Data preparation

The values of the columns are strings separated by “;”, so we need to use the strSplit() function to obtain all the data in one vector so we can work with it.

## Data exploration and Model Deployment

With Shiny, we put a title explaining what the plot is gonna do, and after that, we simply draw the plot with the data without any input.




The less important data, we combine it into one simple value, named as “Others”.

# Salary expectancy of each language

## Data collection (StackOverflow)

This dataset contains a column “CompTotal” which contains the annual salary of each respondent and “LanguageHaveWorkedWith” which contains a string composed with the name of all languages that the respondent have worked with. It is very difficult to study a variable in the Job Market such as salary because it is very difficult to find samples with workers that have more than 30 years of experience. But using means of salary/exp we can estimate which language is more valued.

I considered only the rows where “CompTotal” and “LanguageHaveWorkedWith” are not NA and separate all outliners, for example: 100$ a year (It is below the minimum of USA so the respondent may have not understood the question). Besides, it considers the exchange of “1$ = 1€” to maximize the sample.

To read the csv file, I used the read.csv file considering that it have header, “,” as separator, “.” for decimal numbers (Even we don't have decimal numbers), “NA” as NA value, fill missing data with NA and marked “ as a quote

## Data preparation

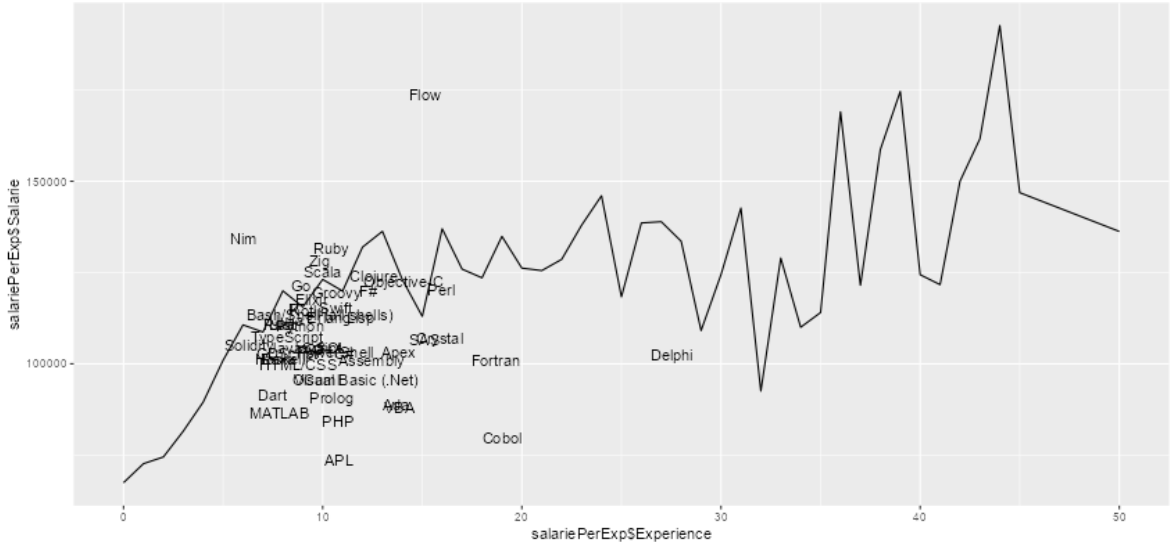
In order to avoid all the distortion in the statistics I did not include currencies that are not a stable currency or currencies that have a small

sample, for example “ARS Argentine peso”(Inflation), or “DKK Danish krone” (sample too small).

To determine the value of each programming language in the job market y had to create another 2 dataframe, on the one hand that contains the mean salary for each year of experience, giving a line plot. On the other hand a dataframe that contains each language with a mean salary of people that have worked with that language and their mean salary giving a dot plot.

## Data exploration and Model Deployment

The objective of this part is to determine how the job market values each language so we can combine the 2 plots that are explained in the previous section. Giving a following result:




Under the line (which is reflects the mean salary) we can found most of languages and if the language is far from the corresponding mean languages we can consider that it is not a demanding language in the job market, if it's under but close to the mean, it means that is a searched knowledge and the value corresponds to the mean not excessively valued but not devalued neither. Above the line follows the same logic, it means that it is a highly valued language.

(This statistics has less precision when it comes to old languages such as delphi or flow, most of these languages are educational or used by people who have more year experience so probably working as senior or freelancer so the mean salary of them must be much higher or lower than the real mean, besides the sample is smaller, so it doesn’t mean that the language itself has a high value)

# Popular IDEs for each language

## Data collection (StackOverflow)

The StackOverflow dataset is a big dataset that contains answers over different topics regarding the world of programming from the Stack Overflow yearly survey. In order to make the dataset more manageable, I used pandas to drop the less significant rows and columns from it, making it considerably lighter.

In order to get the data for the popular IDEs I used the LanguagesHaveWorkedWith column, which contains the languages each contestant has worked with in order, from most to least used. The other relevant column was the NEWCollabToolsHaveWorkedWith, which contains a string with the different development environments each developer has worked with, separated by semicolons. In this case the IDEs are just ordered alphabetically, so there is no meaningful information in the order.

To read the csv file, I used the read.csv file considering that it have header, “,” as separator, “.” for decimal numbers (Even we don't have decimal numbers), “NA” as NA value, fill missing data with NA and marked “ as a quote.

## Data preparation

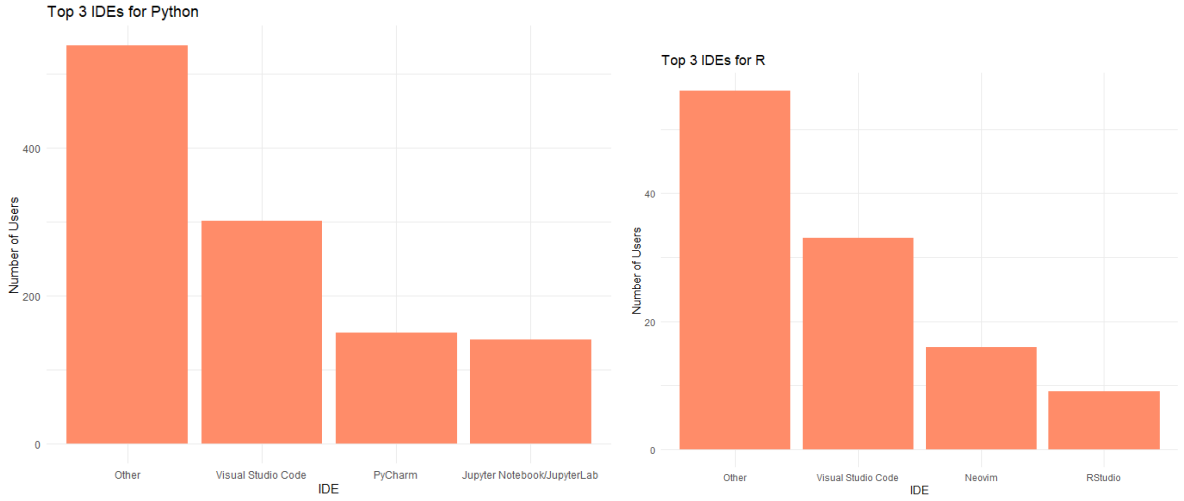
In order to determine the language each respondent works with, I determined that the best results were achieved by picking the first language in the list with str_stract and “[^;]+ ” regular expressions. Then I extracted all the different IDE2 used by each respondent with strsplit() and used a table to store the count of each IDE for each language.

Then I repeated this process over all the unique languages, and finally stored them in a named list of dataframes containing the IDEs and their count.

## Data exploration and Model Deployment

If you look at the shiny app you can see a tabbed panel that allows you to visualize the most used IDEs of the users that use each language. Since the actual number of IDE’s used for each language is quite large, I considered

that only the 3 most used IDEs were really relevant and added the rest into the “other” column.




With these plots, we can get to the realization that there isn’t such variability in terms of IDE usage between programming languages (And this variability was even lower if we considered all the languages each user has worked with). This is true because some IDEs like Visual Studio Code or Vim are not language specific IDEs. Nonetheless we can still get interesting information from the second and third IDEs from each language.

# Employability of each language

## Data collection (StackOverflow)

The StackOverflow dataset is a big dataset that contains answers over different topics regarding the world of programming from the Stack Overflow yearly survey. In order to make the dataset more manageable, I used pandas to drop the less significant rows and columns from it, making it considerably lighter.

In order to get the employment data, I used two different metrics from the dataset. One coming from the DevType column, and the other one from the EdLevel Column. Of course to get the employment state I used the Employment column in the same dataset.

To read the csv file, I used the read.csv file considering that it have header, “,” as separator, “.” for decimal numbers (Even we don't have decimal numbers), “NA” as NA value, fill missing data with NA and marked “ as a quote.

## Data preparation

As I just mentioned, I measured employment in relation to two different metrics, education level and developer type. The approach for both of them was the same just using different columns from the dataset.

For this plot, I just considered the "Employed, full-time" respondents, since considering other employment types would lead to too much variability and outliers. Then for each full-time employed respondent we added their education level and dev type to each respective count in two dataframes.

In the case of the developer types, I decided to just use the top 15 types since the lower ones were just too low. For the devType dataframe I also decided to drop the "Developer" and "Other" categories since they were too broad and didn't give relevant information.

## Data exploration and Model Deployment

For the plots, I decided to display the percentages of each education level and dev type in two separate horizontal bar plots, since the bin count was quite high.