# R: A Statistical Programming Language

Tessa Gaynor, Joel Joseph, Jared Sanchez

# History

- Created by two professors at the University of Auckland in 1992 in a research project
  - Ross Ihaka and Robert Gentleman
- First version released in 1995
- Goal was to be a program to teach in introductory statistics classes
- Professors wanted to create an open source environment that was flexible and accessible for statistical computing and data analysis
- Inspired by the languages S and Scheme

# Overview

- Language for statistical computing and data visualization
- Used for data mining, bioinformatics, and data analysis
- R is a domain specific language (DSL)
  - Specialized for a specific task
- Considered a high level language
- Open source
- Can be used on multiple third party interfaces such as:
  - Rstudio
  - Jupyter

# Design Goals

- Ihaka and Gentleman aimed for an intuitive language that allows users to quickly perform tasks like plotting and summarization; catering to those who need basic functionality without steep learning curves.
- R was designed with a focus on efficiency in data manipulation and analysis. It allows for the creation of complex data structures, such as matrices, data frames, and lists, which are essential for handling large datasets.
- R prioritizes flexibility and extendibility for package developers, allowing users to plug and play with various libraries to expand R’s capabilities.
- R combines lazy functional features with object-oriented programming.

# Language Features

- R handles Memory Management through a unique form of Automatic Garbage Collection. Memory can also be handled semi-manually.
- R uses Pass-by-Value semantics, though there are exceptions where it serves functionally as Pass-by-Reference.
- R supports abstraction through its object-oriented programming (OOP) features. It allows for the creation of objects, which are instances of classes, and the definition of methods that operate on these objects.
    - First-class objects that can be assigned to variables, passed as arguments, and returned from other functions.
    - Functions can take other functions as arguments and return them as output.
    - Limited support through ‘S3’ and ‘S4’ classes for defining object-oriented structures.

# Real-World Applications Part 1

- Data science and big data
  - Google, Facebook, Bing
  - Accenture and DataBricks
- Finance
  - Deloitte and ANZ
- Banking
  - American Express and JP Morgan
- Research and academics
  - Cornell and UCLA
- IT
  - IBM and Oracle

# Real-World Applications Part 2

- E-commerce
  - Amazon and Walmart
- Healthcare
  - Novartis and Public Health Agency of Canada
- Marketing
  - Accenture and Ford
- Government
  - FDA and NASA
- Social media
  - Meta and X

# Sample Code – Data Science example

```
# Define a function to calculate the average
calculate_average <- function(numbers) {
  return(mean(numbers))
}

# Create a vector of numbers
numbers <- c(1, 2, 3, 4, 5)

# Use the function to calculate the average
average <- calculate_average(numbers)

# Print the average
print(average)

# Fit a linear model
model <- lm(numbers ~ 1)

# Print the model summary
print(summary(model))
```

# Sample Code - Finance Example

```
# Define a function to calculate the ROI

calculate_roi <- function(initial_investment, earnings) {

  if (is.numeric(initial_investment) && is.numeric(earnings)) {

    roi <- (earnings - initial_investment) / initial_investment * 100

    return(roi)

  } else {

    stop("Both initial investment and earnings must be numeric.")

  }

}

# Define the initial investment and the earnings

initial_investment <- c(1000, 2000, 3000, 4000, 5000)

earnings <- c(1200, 2400, 3600, 4800, 6000)
```

```
# Use the function to calculate the ROI

roi <- sapply(1:length(initial_investment), function(i) calculate_roi(initial_investment[i], earnings[i]))


# Print the ROI

print(roi)


# Fit a linear model to predict ROI based on initial investment

model <- lm(roi ~ initial_investment)


# Print the summary of the model

print(summary(model))
```

# Sample Code - Government Example

```
# Define a function to find the maximum population
find_max_population <- function(population) {
  if (is.vector(population) && is.numeric(population)) {
    max_population <- max(population)
    return(max_population)
  } else {
    stop("Input must be a numeric vector.")
  }
}

# Create a vector of population numbers
population <- c(100000, 102000, 104040, 106121, 108365)

# Use the function to find the maximum population
max_population <- find_max_population(population)

# Print the maximum population
print(max_population)

# Define the time periods
time <- 1:length(population)
# Fit an exponential growth model to the population
model <- nls(population ~ a * exp(b * time), start = list(a = 100000, b = 0.01))
# Print the summary of the model
print(summary(model))
```

# Next Steps for R Part 1

- Performance and memory management
- User friendly syntax
- Better integration with other languages
- Improved error handling

# Next Steps for R Part 2

- Enhanced visualization libraries
- More comprehensive documentation
- Better support for web applications
- Expanded machine learning capabilities


Demo Time

# References


Agrawal, V. (2021, May 12). *Applications of R programming in R-EAL world*. eLearning Industry. https://elearningindustry.com/applications-r-programming-r-eal-world

Bates, C. (2023, April 9). *Learn R the right way in 5 steps (2021 update)*. Dataquest. https://www.dataquest.io/blog/learn-r-for-data-science/

Gour, R. (2019, May 13). *How R is used in Data Science (13 real-life analogies)*. Medium. https://codeburst.io/how-r-is-used-in-data-science-13-real-life-analogies-3f379de5e8ec

*Memory*. Memory usage · Advanced R. (n.d.). http://adv-r.had.co.nz/memory.html#:~:text=R%20uses%20an%20alternative%20approach,object%2C%20it%20deletes%20that%20object.

Mount, G. (2024, April 15). *Evaluating & Optimizing Code performance in R: Master data skills + AI*. Master Data Skills + AI | Insights and Strategies from the Enterprise DNA Blog. https://blog.enterprisedna.co/evaluating-optimizing-code-performance-in-r/

Talreja, S. (2023, July 18). *Tips and tricks for efficient coding in R - dzone*. dzone.com. https://dzone.com/articles/tips-and-tricks-for-efficient-coding-in-r

Team, C. (2024, March 28). *9+ best applications of R: Why do companies use it?*https://www.calltutors.com/blog/applications-of-r/

Vidvan, T. (2019, December 19). *R applications - 9 real-world use cases of R programming*. TechVidvan. https://techvidvan.com/tutorials/r-applications/

Worsley, S. (2023, October 17). *What is R? - an introduction to the statistical computing powerhouse*. DataCamp. https://www.datacamp.com/blog/all-about-r#