# After the Exam

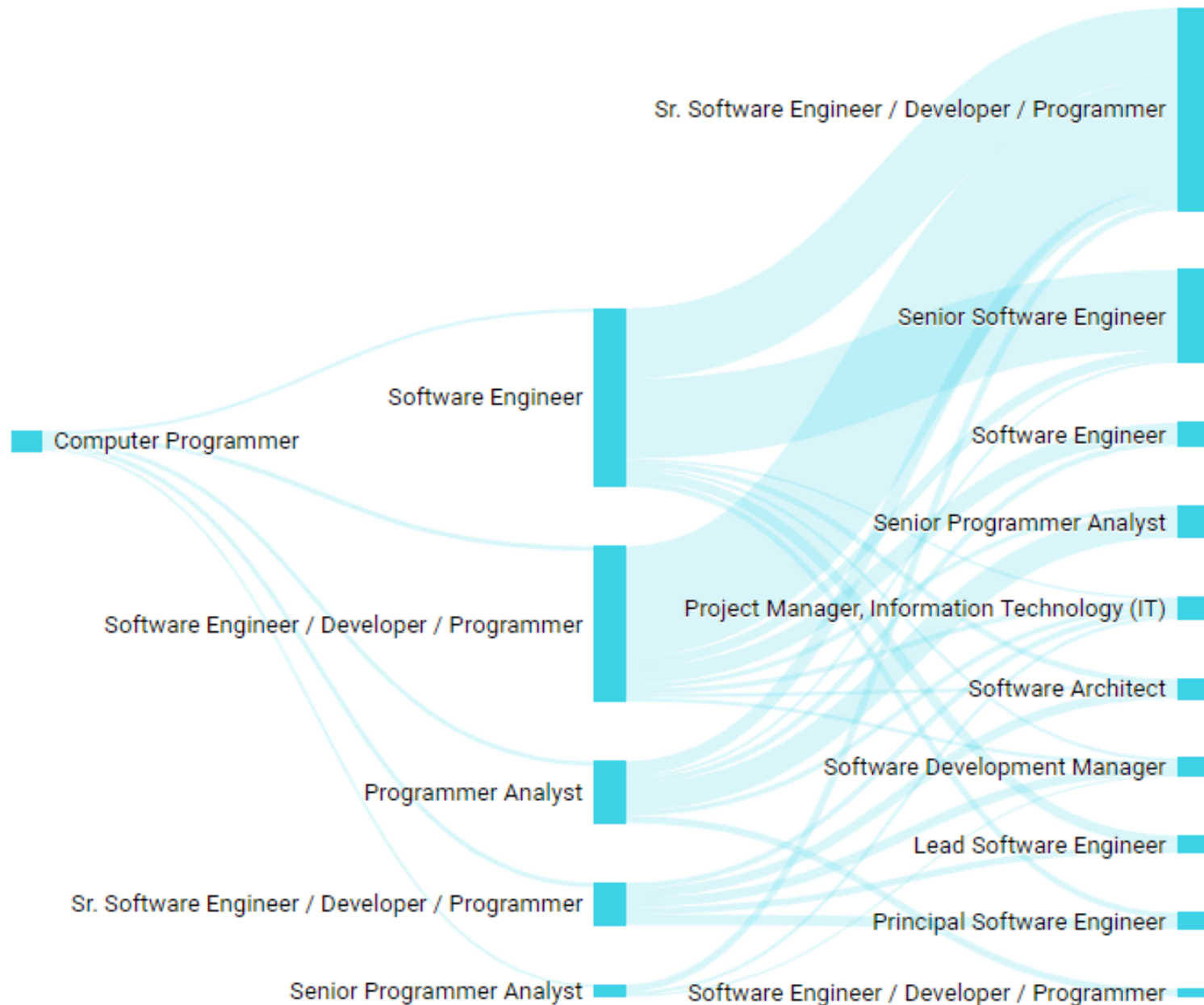# Group or Individual Project

- Today
  - Look at options/Research/What are good languages to learn, potential computer science career paths,…
  - Select a project (Team or individual)
  - Set up and share Google Slides
- Tuesday:
  - Set sprint and work on project
- Thursday…:
  - Continue Sprint
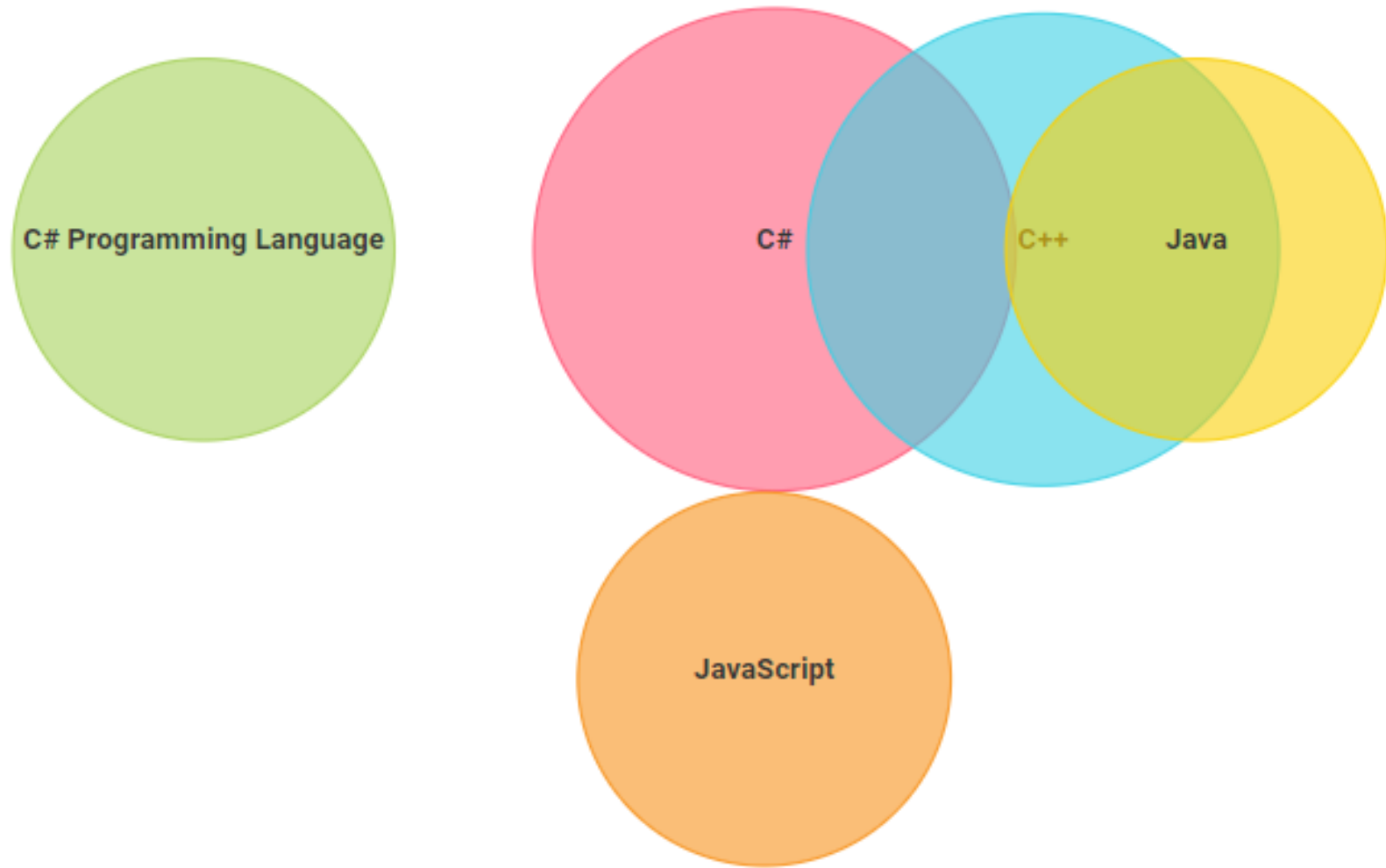- Your final exam is your Google Slides presentation.

# Some Project Options

- Study a new language and present the basic constructs to the class.
- Develop a project
- Develop a game in any language
- Develop an App for your phone/…
- Study and present on a new language
- Study and present Other Integrated Development Environments.
- Describe your language/project you have been working on
- Show an example
- The next few slides are to look at the computer science

# Common Career Paths for Computer Programmer



Sr. Software Engineer / Developer / Programmer

Senior Software Engineer

Software Engineer

Software Engineer

Senior Programmer Analyst

Project Manager, Information Technology (IT)

Computer Programmer

Software Architect

Software Engineer / Developer / Programmer

Software Development Manager

Programmer Analyst

Lead Software Engineer

Sr. Software Engineer / Developer / Programmer

Principal Software Engineer

Senior Programmer Analyst

Software Engineer / Developer / Programmer

# Popular Skills for Computer Programmer

C# Programming Language

C#

C++

Java

JavaScript

*This chart shows the most popular skills for this job and what effect each skill has on pay.*

# Popularity of Programming Languages

## TIOBE Programming Community Index for May 2020

| May 2020 | May 2019 | Change | Programming Language |
|----------|----------|--------|----------------------|
| 1 | 2 | ⌃ | C |
| 2 | 1 | ⌄ | Java |
| 3 | 4 | ⌃ | Python |
| 4 | 3 | ⌄ | C++ |
| 5 | 6 | ⌃ | C# |
| 6 | 5 | ⌄ | Visual Basic |
| 7 | 7 |  | JavaScript |
| 8 | 9 | ⌃ | PHP |
| 9 | 8 | ⌄ | SQL |
| 10 | 21 | ⌃⌃ | R |
| 11 | 18 | ⌃⌃ | Swift |
| 12 | 19 | ⌃⌃ | Go |
| 13 | 14 | ⌃ | MATLAB |
| 14 | 10 | ⌄⌄ | Assembly language |
| 15 | 15 |  | Ruby |
| 16 | 20 | ⌃⌃ | PL/SQL |
| 17 | 16 | ⌄ | Classic Visual Basic |
| 18 | 13 | ⌄⌄ | Perl |
| 19 | 28 | ⌃⌃ | Scratch |
| 20 | 11 | ⌄⌄ | Objective-C |

| Position | Programming Language |
|----------|----------------------|
| 21 | Rust |
| 22 | Dart |
| 23 | Delphi |
| 24 | D |
| 25 | COBOL |
| 26 | Kotlin |
| 27 | Transact-SQL |
| 28 | Scala |
| 29 | SAS |
| 30 | ABAP |
| 31 | Lisp |
| 32 | Lua |
| 33 | Ada |
| 34 | Fortran |
| 35 | PowerShell |
| 36 | Logo |
| 37 | VBScript |
| 38 | Groovy |
| 39 | Julia |
| 40 | LabVIEW |
| 41 | Haskell |
| 42 | ActionScript |
| 43 | Scheme |
| 44 | TypeScript |
| 45 | Apex |
| 46 | F# |
| 47 | Prolog |
| 48 | OpenEdge ABL |
| 49 | RPG |
| 50 | Erlang |

# Most In Demand Languages

Here are the Top 7 programming languages with most job posting on Indeed as of January 2019:
- **Java – 65,986 jobs**
- **Python – 61,818 jobs**
- **Javascript – 38,018 jobs**
- **C++ – 36,798 jobs**
- **C# – 27,521 jobs**
- **PHP – 16,890 jobs**
- **PERL – 13, 727 job**

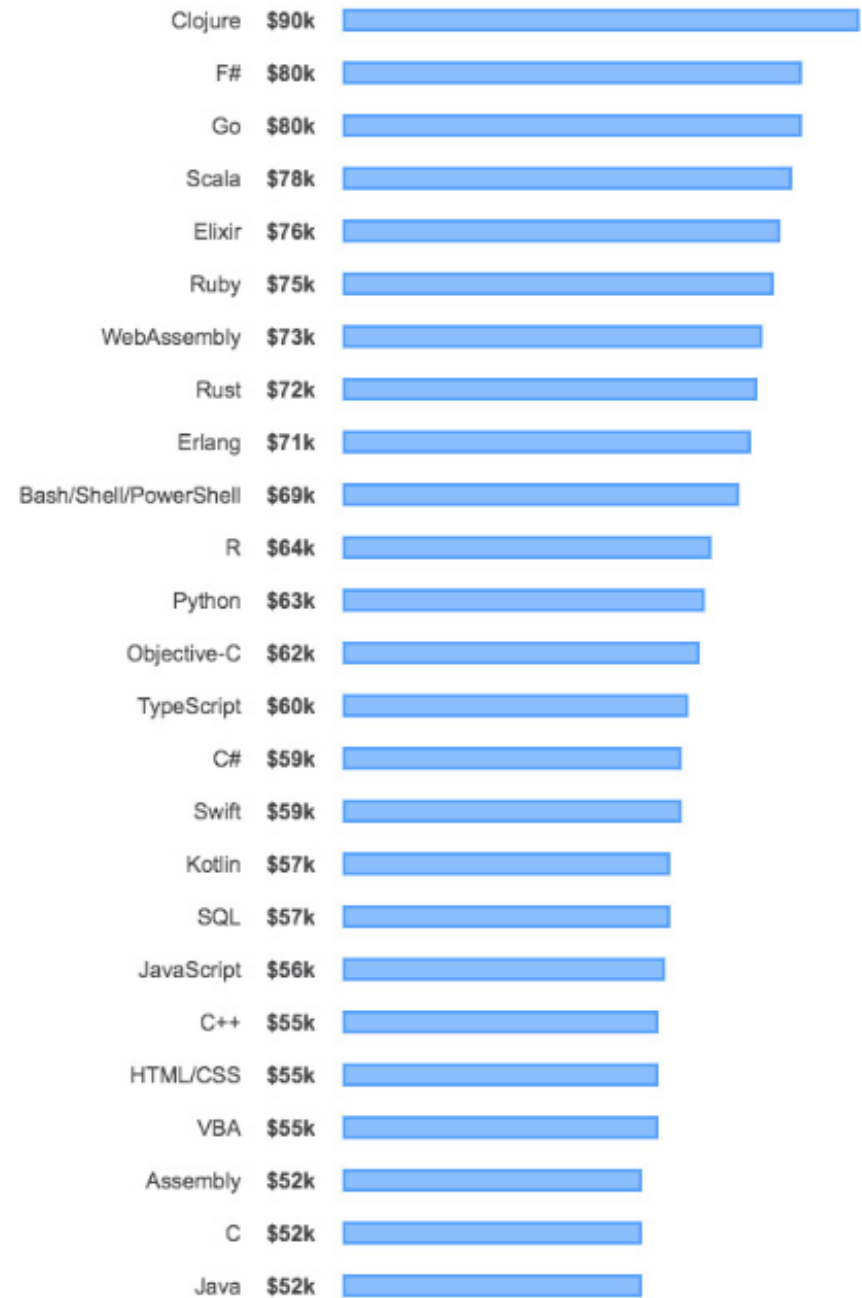# THE TEN BEST PROGRAMMING LANGUAGES TO LEARN IN 2020. HACKR.IO

1. Python
2. Java
3. C/C++
4. JavaScript
5. **Golang** programming language: By Google
6. **R** : For Data Analysis and Machine Learning
7. **Swift**: For developing iOS applications
8. **PHP**: Back end programming,
9. C#
10. **MATLAB**: Statistical analysis tool.  Used in engineering, computer cision and image processing.

https://hackr.io/blog/best-programming-languages-to-learn-2020-jobs-future

# Highest Paid Programming Languages

**Highest-paid programming and scripting languages in 2018**

| Language | Average salary | YoY % Increase |
|---|---|---|
| 1.Golang | $132,827 | N/A |
| 2. Perl | $110,678 | -2.7 |
| 3. Shell | $109,518 | 4.0 |
| 4. Node.js JavaScript | $105,418 | 3.7 |
| 5. Java/J2EE | $105,164 | 2.6 |
| 6. TypeScript | $103,680 | 1.7 |
| 7. Python | $103,587 | 0.4 |
| 8. Ruby | $ 102,086 | -1.2 |
| 9. Swift | $ 101,631 | 3.2 |
| 10. C# | $101,566 | 1.8 |

# Lvivity: Top Highly Paid Programming Languages to Learn in 2019

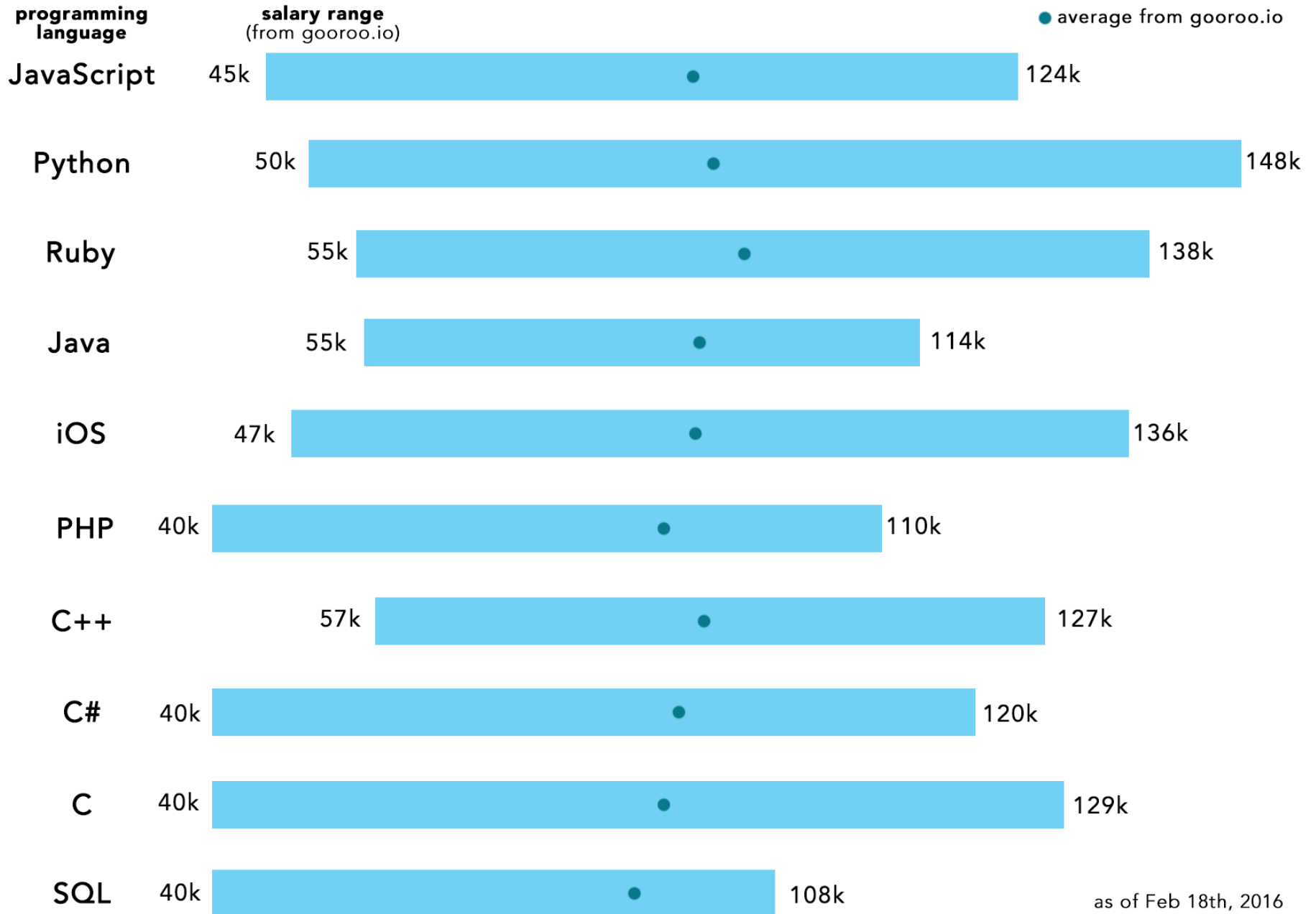| | |
|---|---|
| Clojure | $90k |
| F# | $80k |
| Go | $80k |
| Scala | $78k |
| Elixir | $76k |
| Ruby | $75k |
| WebAssembly | $73k |
| Rust | $72k |
| Erlang | $71k |
| Bash/Shell/PowerShell | $69k |
| R | $64k |
| Python | $63k |
| Objective-C | $62k |
| TypeScript | $60k |
| C# | $59k |
| Swift | $59k |
| Kotlin | $57k |
| SQL | $57k |
| JavaScript | $56k |
| C++ | $55k |
| HTML/CSS | $55k |
| VBA | $55k |
| Assembly | $52k |
| C | $52k |
| Java | $52k |

# GeeksforGeeks

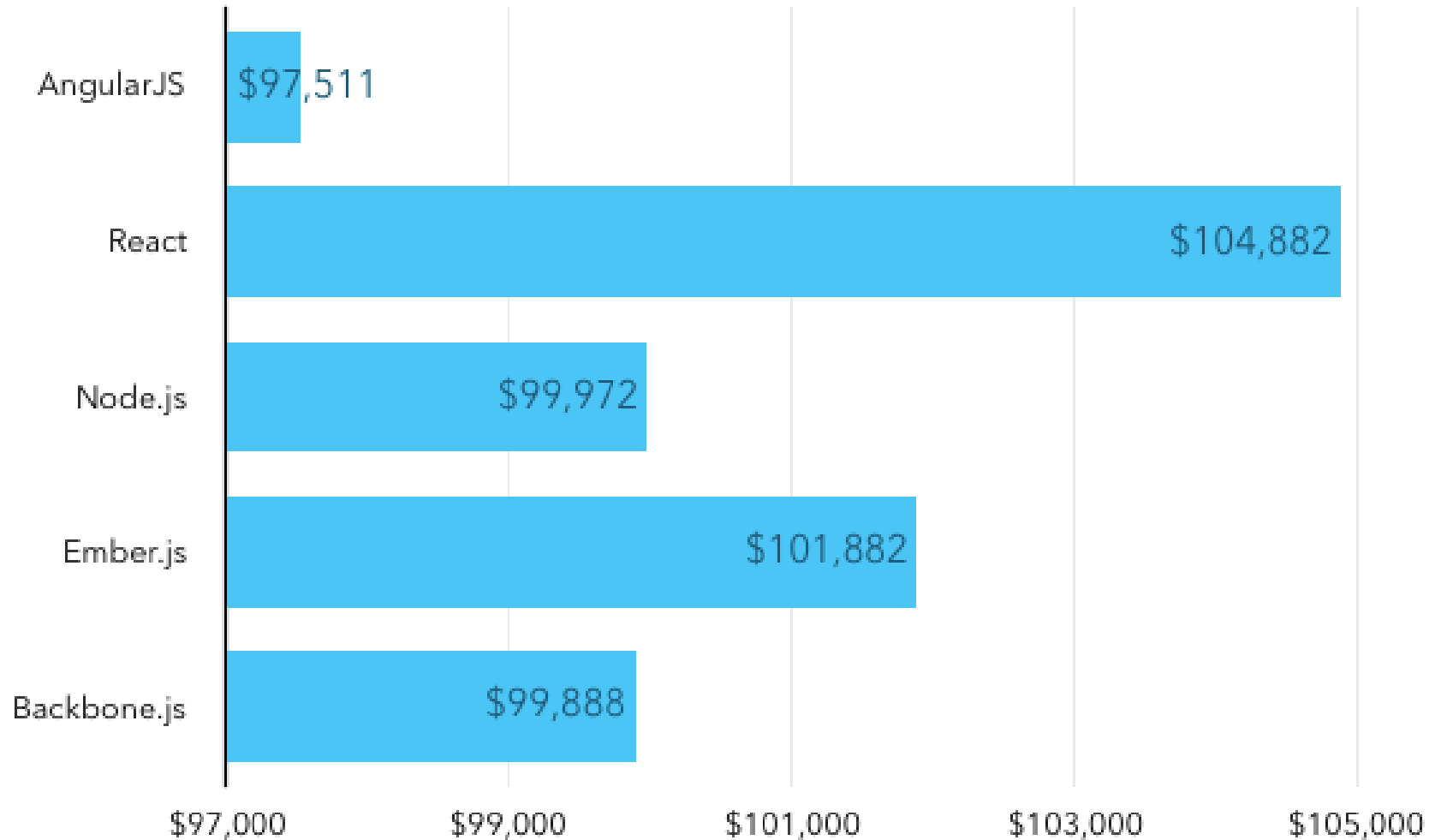Once again, here is the list of top 10 programming languages to start with:
1. JavaScript
2. Python
3. Java
4. C/CPP
5. PHP
6. Swift
7. C#
8. Ruby
9. Objective – C
10. SQ

# 2016 Programming Salary Range in the US



**programming language**     **salary range** (from gooroo.io)     ● average from gooroo.io

| language | min | max |
|---|---|---|
| JavaScript | 45k | 124k |
| Python | 50k | 148k |
| Ruby | 55k | 138k |
| Java | 55k | 114k |
| iOS | 47k | 136k |
| PHP | 40k | 110k |
| C++ | 57k | 127k |
| C# | 40k | 120k |
| C | 40k | 129k |
| SQL | 40k | 108k |

as of Feb 18th, 2016
graph by codementor.io

# JavaScript Frameworks

## Average Salary from Gooroo.io as of February 2016 (USD)

| Framework | Salary |
|-----------|--------|
| AngularJS | $97,511 |
| React | $104,882 |
| Node.js | $99,972 |
| Ember.js | $101,882 |
| Backbone.js | $99,888 |

$97,000  $99,000  $101,000  $103,000  $105,000

# Examples

- Languages
  - C++
  - Scheme
  - C#
  - Python
  - Alice
  - BYOB
  - MATLab
- Robotics
  - V5 Coding Studio
  - VEXCode Virtual World (https://vr.vex.com/)
  - RobotMesh Studio Mimic (https://www.robotmesh.com/studio)
  - RobotC: PID Control: Driving straight, with potentiometer for an arm to reach a certain position
  - MATLab
  - Onshape Free web based Computer Aided Design (CAD)

There are CodeHS courses on many of these languages I can set up for you.

Java Topics
Files,
Stacks/Queues/Linked Lists
Trees
Hashing (Arrays/Files)
Heaps

# Grading:

- Daily Journal (PowerPoint) updates. Four points per day
- Sprint Backlog: What you commit to complete in the next Sprint (10 Points/Sprint)
- Sprint Review/Demonstrate what is done:
- Sprint Retrospective: Final
  - 10 Points

# Individual Documenting your Work

- Google Slides to Document your progress
- Page One: Title Page
  - Your name, type of project
- Page Two: Sprint Backlog
  - Tasks you intend to get done during the next Sprint.
- Scrum Meeting
  - Each day each person records…
  - What was accomplished yesterday (last class).
  - What you intend to accomplish today.
  - What hindrances are getting in your way.
- Sprint Review
  - Done at the end of the sprint. A demonstration of what has been 'done'.
- Sprint Retrospective
  - Done at the end of the sprint. Describe what went well, what did not go well and potential improvements.

# Presentations

- The presentations will be the final exam for this course
  - Describe your language/project you have been working on
  - Show an example
  - +/- 20 minutes per presentation/activity
- Participants will be evaluating the presentations