Compilers

2017

Outline

- Language Popularity
- Introduction
- History
- Compiler Structure
- References

Where they came from?

1954: IBM Develops the 704 – Assembly code

Fortran I: John Backus (later Algol and LISP)

- High Level Code to Assembly
- 1958: 50% software is in Fortran!
- First Compiler!

Language's popularity

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9

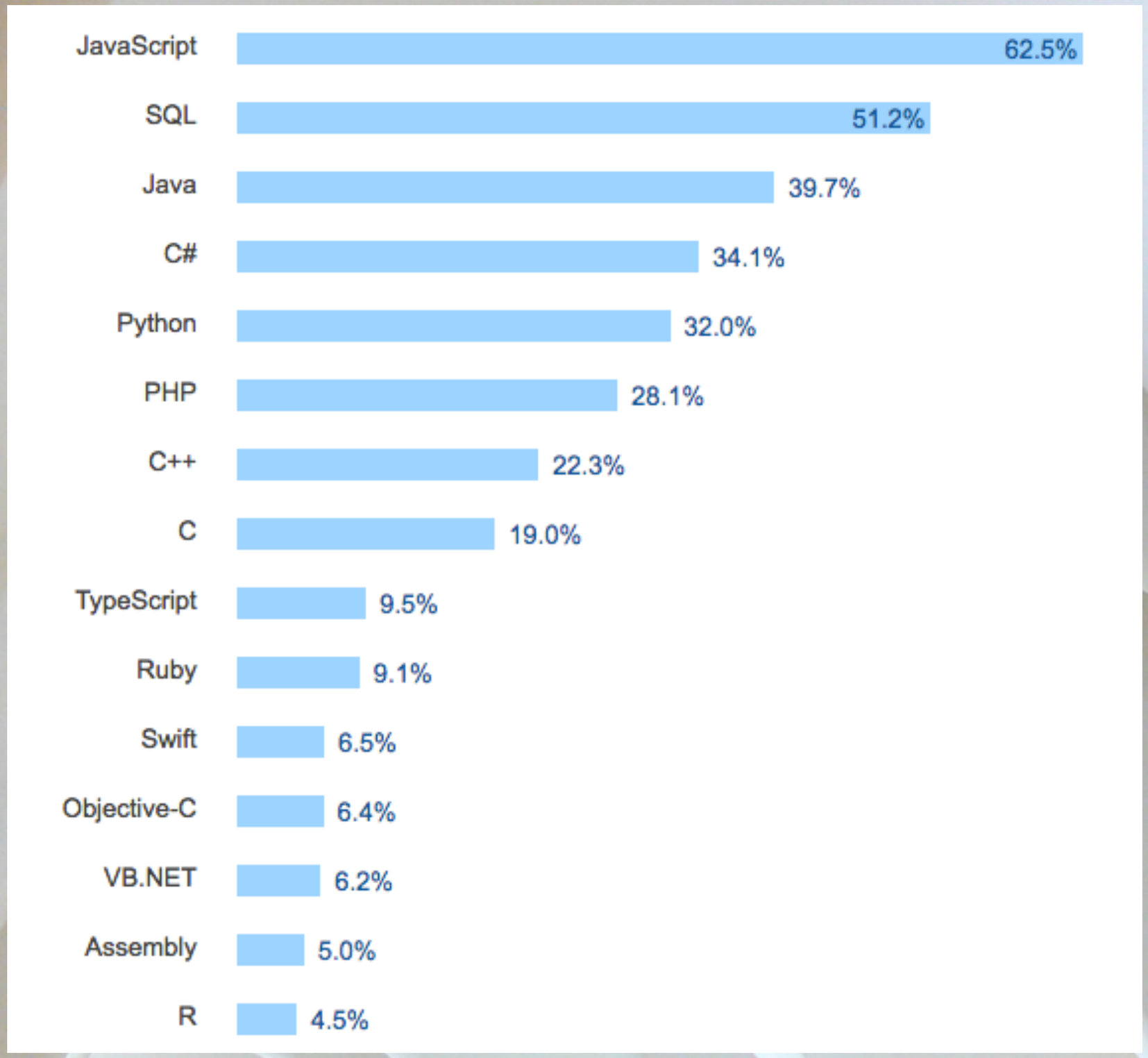
IEEE Spectrum Language Ranking 2016

Language's popularity

Language Rank	Types	Spectrum Ranking
1. Python		100.0
2. C		99.7
3. Java		99.5
4. C++		97.1
5. C#		87.7
6. R		87.7
7. JavaScript		85.6
8. PHP		81.2
9. Go		75.1
10. Swift		73.7

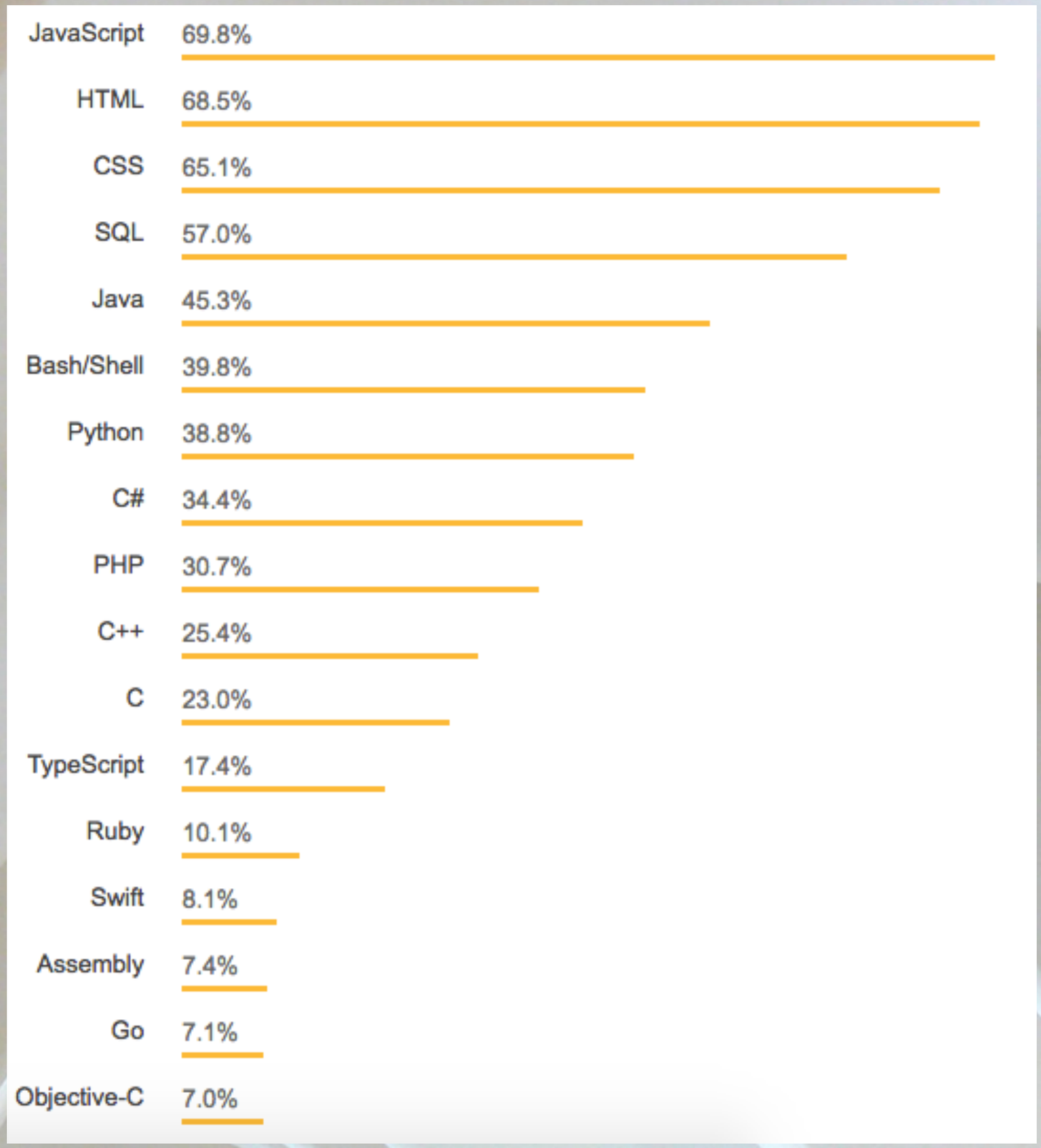
IEEE Spectrum Language Ranking 2017

Language's popularity



Stack Overflow Survey 2017

Language's popularity



Stack Overflow Survey 2018

Language's popularity

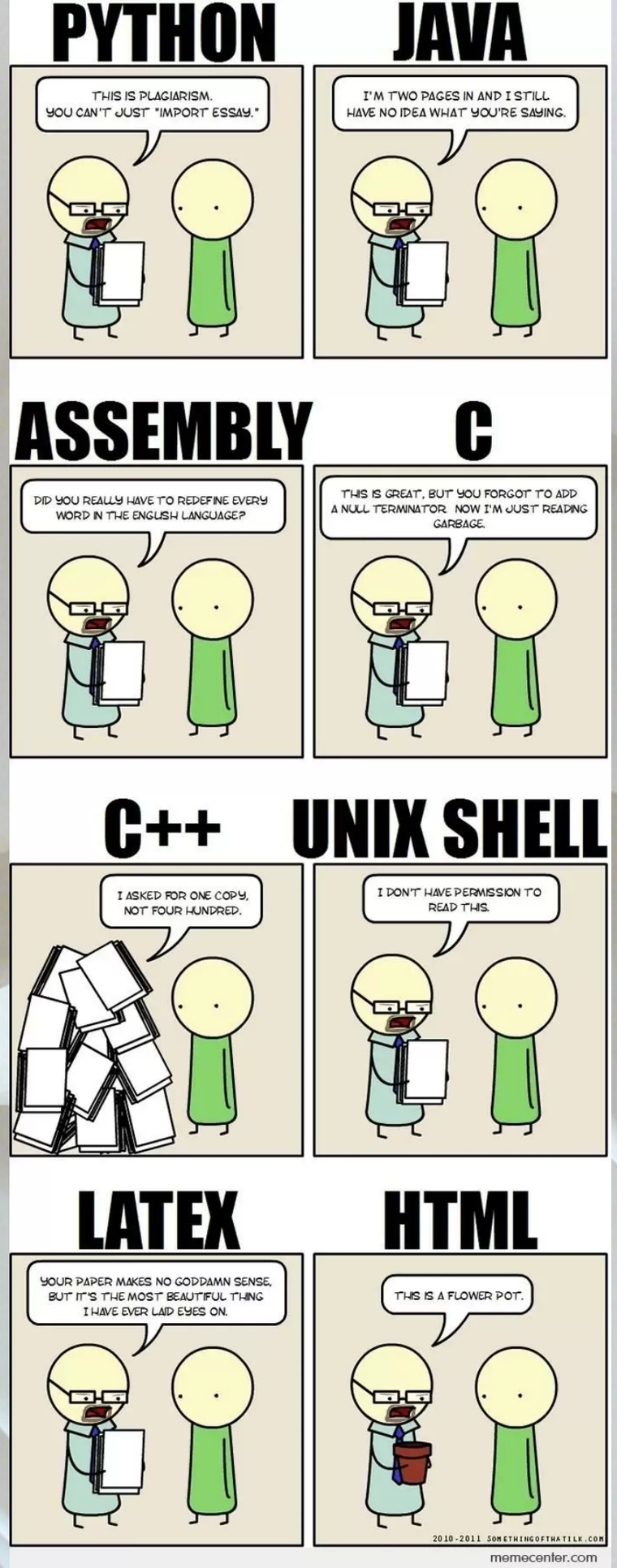
TIOBE INDEX 2016

Apr 2018	Apr 2017	Change	Programming Language	Ratings	Change
1	1		Java	15.777%	+0.21%
2	2		C	13.589%	+6.62%
3	3		C++	7.218%	+2.66%
4	5	^	Python	5.803%	+2.35%
5	4	v	C#	5.265%	+1.69%
6	7	^	Visual Basic .NET	4.947%	+1.70%
7	6	v	PHP	4.218%	+0.84%
8	8		JavaScript	3.492%	+0.64%
9	-	^^	SQL	2.650%	+2.65%
10	11	^	Ruby	2.018%	-0.29%
11	9	v	Delphi/Object Pascal	1.961%	-0.86%
12	15	^	R	1.806%	-0.33%
13	16	^	Visual Basic	1.798%	-0.26%
14	13	v	Assembly language	1.655%	-0.51%
15	12	v	Swift	1.534%	-0.75%
16	10	vv	Perl	1.527%	-0.89%
17	17		MATLAB	1.457%	-0.59%
18	**14**	vv	**Objective-C**	**1.250%**	**-0.91%**

Language Design

- Imperative / Declarative
- Procedural / OOP
- Functional
- Constraint
- Logic
- von Neumann
- 4GL, Event-driven, Parallel, etc, etc, etc

Language Design



Language Design

- Evolution
- Personal Taste
- Special Purposes
- Pedagogy
- Standardization
- Open Source-ness
- Patronage

Turing-Complete Languages

Language Design

The art of telling another human being what one wants the computer to do.

Donald Knuth

Language Design

- What makes a programming language successful?
- Why are there so many programming languages?
- Which one would you like or prefer ?

Introduction to Translation

Compilers

The whole picture

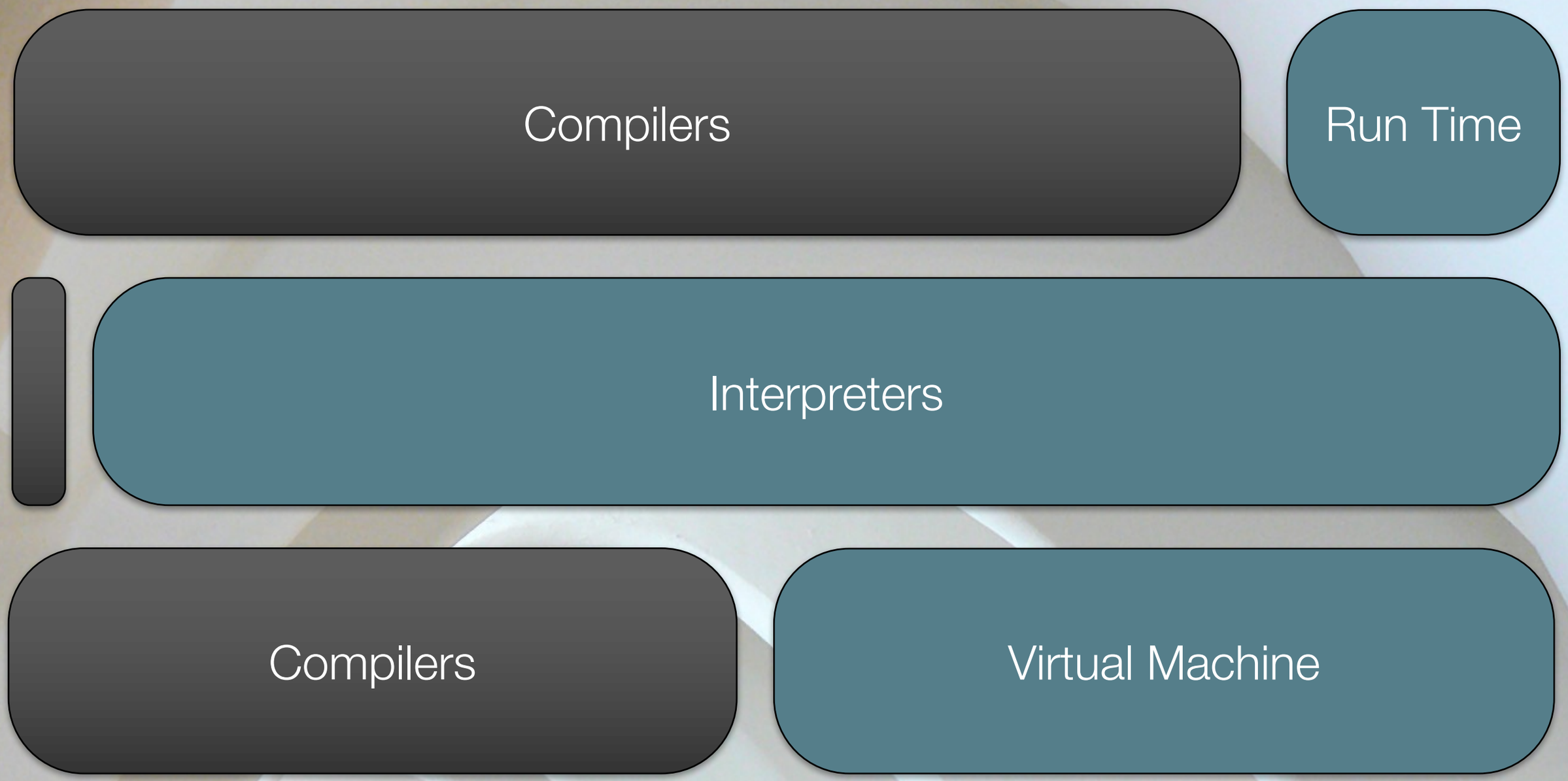
Extensive Preprocessing

Interpreters

Line by line

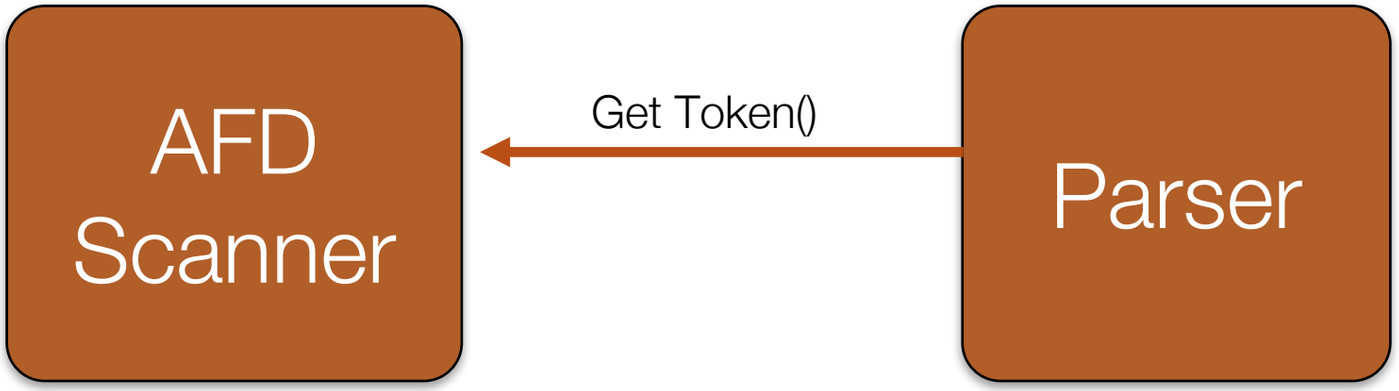
Run programs as they are

Introduction to Translation



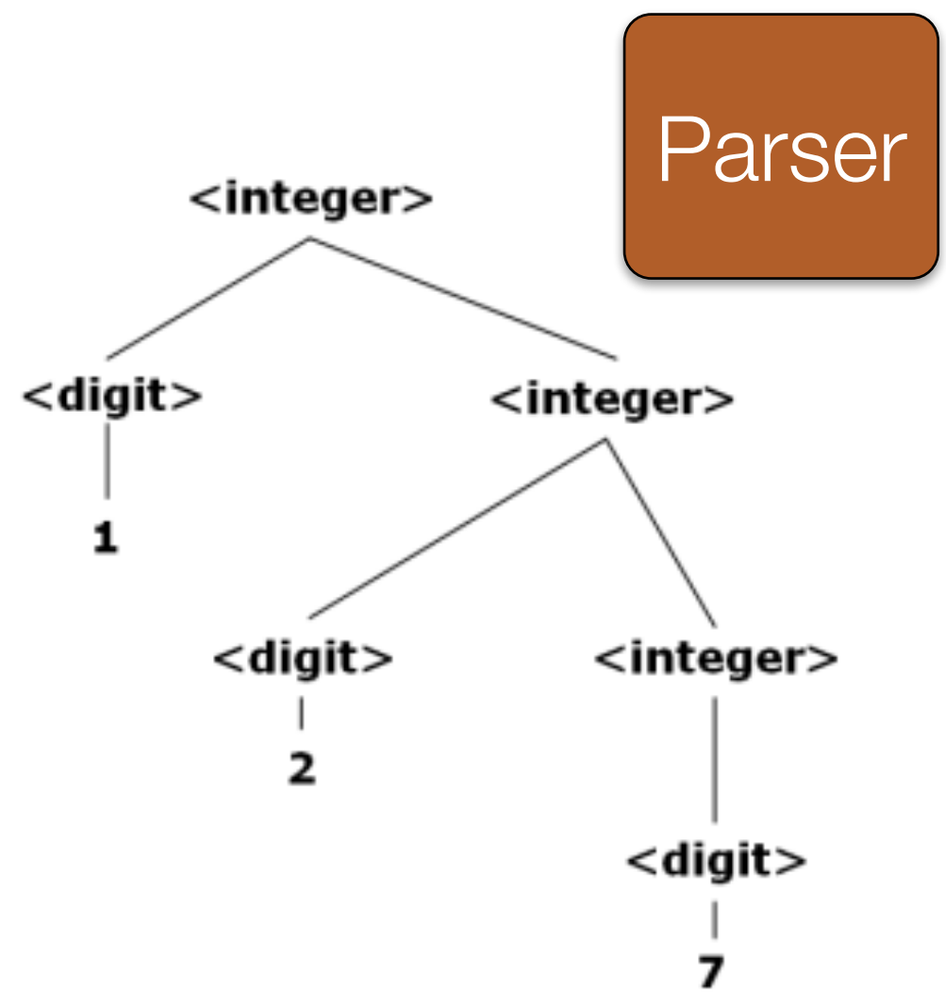
Compiler Structure

1. **Lexical Analysis**
2. **Parsing**
3. Semantic Analysis
4. Optimization
5. Code Generation



Compiler Structure

1. Lexical Analysis
2. Parsing
3. **Semantic Analysis**
4. Optimization
5. Code Generation



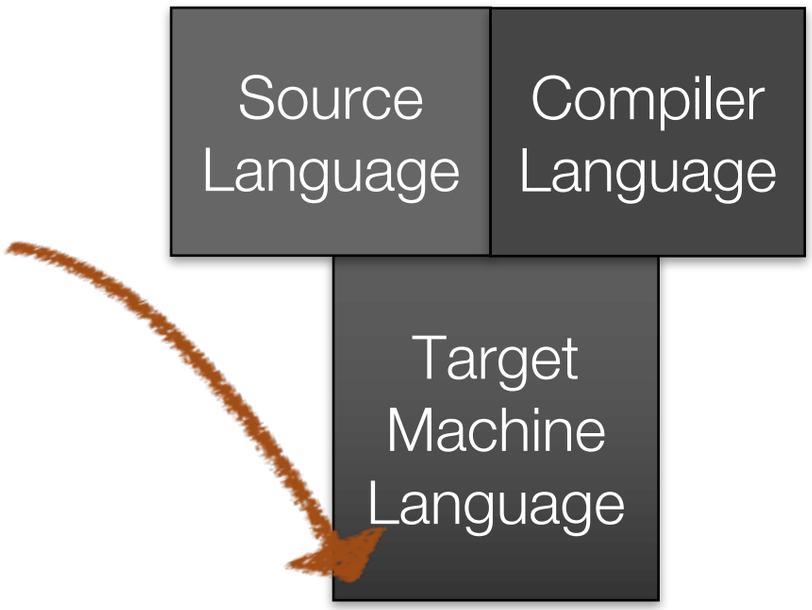
Compiler Structure

1. Lexical Analysis
2. Parsing
3. Semantic Analysis
4. **Optimization**
5. Code Generation

```
int X = 3 + 7;
```

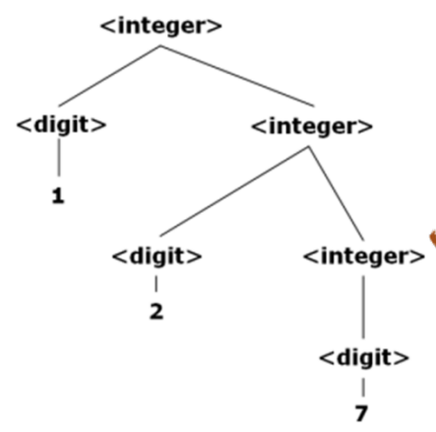
↓

```
int X = 10;
```



Compiler Structure

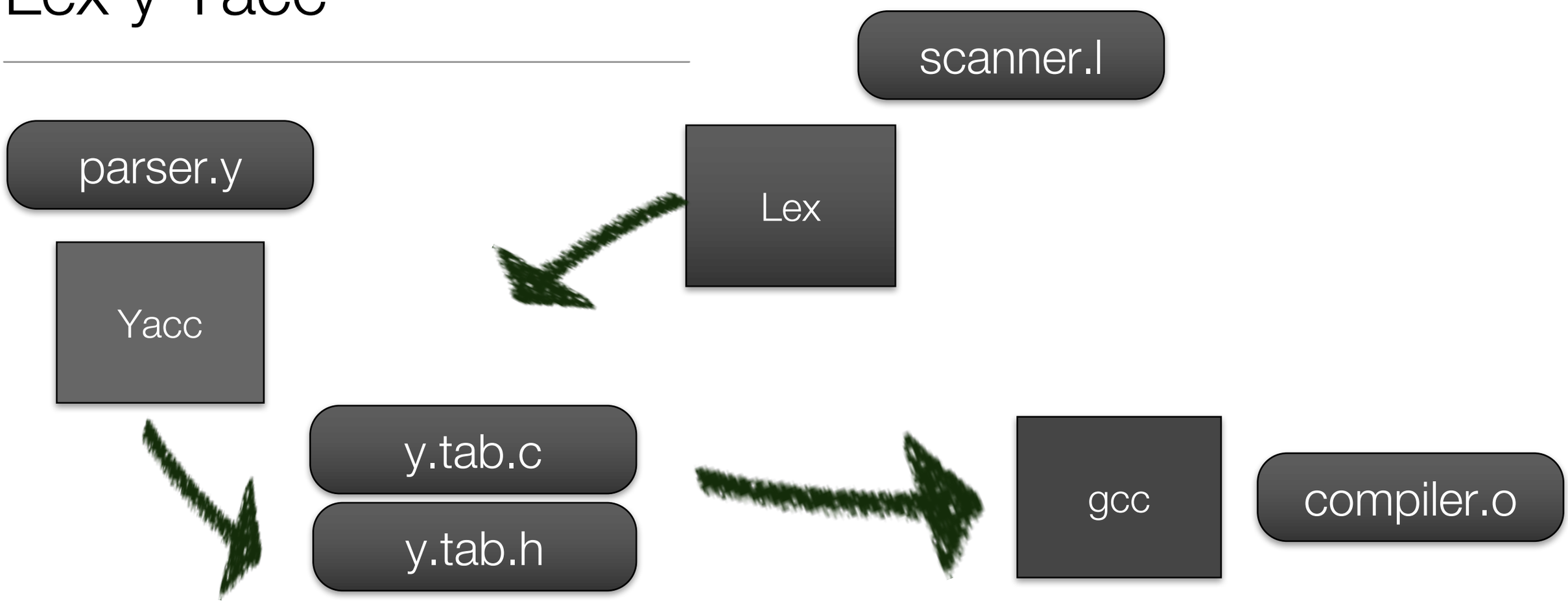
1. Lexical Analysis
2. Parsing
3. Semantic Analysis
4. Optimization
5. **Code Generation**



IL: Intermediate Code

```
0400 2073FE JSR $FE73        s~
0403 A200   LDX #$0         "□
0405 BD8004 LDA $480,X      =□↖
0408 F006   BEQ $410        p✓
040A 2075FE JSR $FE75        u~
040D E8     INX             h
040E D0F5   BNE $405        Pu
0410 00     BRK             □
0411 B9     *=$480
0480 48     'H              H
0481 45     'E              E
0482 4C     'L              L
0483 4C     'L              L
0484 4F     'O              O
0485 00     $0              □
0486 67     !
```

Lex y Yacc



Components

- Semantic Analysis: attribute grammars
- Symbols Table
- Runtime Environment: Memory Management
- Abstract Syntax Tree
- Intermediate Language: IL , Abstract Stack Machine, Three Address Codes
- Target Language: Assembler, Object
- Linking

References

- Dragon Book (Aho, Sethi, Ullman)
- Compiladores, Teoría e Implementación (Jacinto Catalán)
- http://www.tldp.org/LDP/LGNET/issue41/sevenich.html
- Programming Languages Pragmatic, Scott, 2009
- Practical Foundations for Programming Languages, Harper,2