
Extra
01
CST8152
Compilers

Algonquin College

Computer Engineering Technology

CST8152 Compilers

Fall, 2023

Extra 01

Prof. Paulo Sousa

Algonquin College

Computer Engineering Technology

CST8152 Compilers

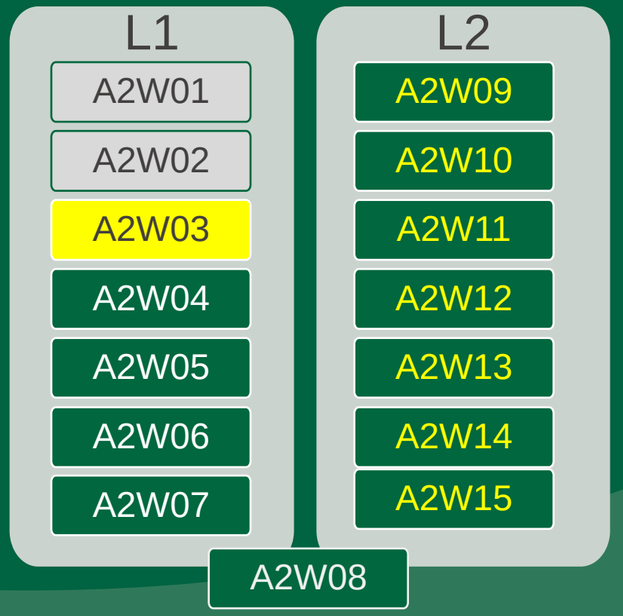
Fall, 2023

Extra 01

Prof. Paulo Sousa

A11-A12: New languages

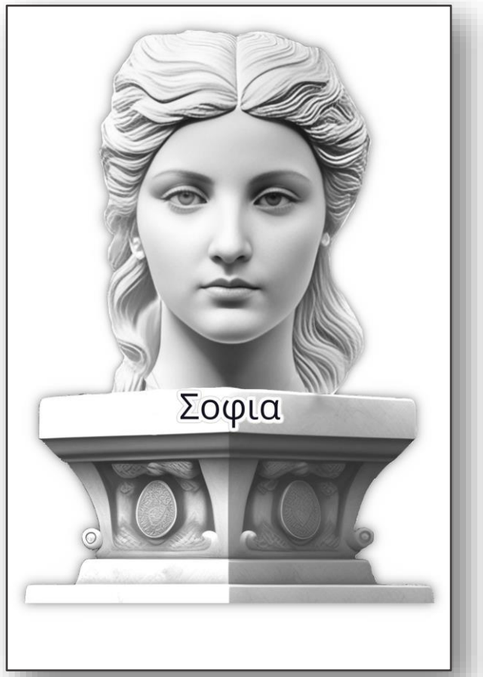
New Compiler Specification



Ex1: Sofia Language

In the Lecture Notes:

- **Appendixes 1-4:**
  - **Appendix 1: Go/DSL** examples
    - Tip: Check similarities and differences with original Python language.
  - **Appendix 2: Sofia** initial models (to be used in A21)
  - **Appendix 3: Sofia** ILS: Informal language spec.
    - Tip: Check how basic elements are defined in the language.
  - **Appendix 3: Sofia** BNF (to be used in A31).



Ex1: Sofia Language

Learning by Example

- **Show your examples:** Anyone (including you) can learn by examples:

```
# Sofia Hello #
  main& {
      data { }
      code {
             print&('Hello World!');
       }
}
```

```
# Sofia Example (Volume of a sphere) #
main& {
      data {
            real PI%, r%, Vol%;
      }
      code {
            PI% = 3.14;
            input&(r%);
            Vol% = 4.0 / 3.0 * PI% * (r% * r% * r%);
            print&(Vol%);
      }
}
```

Compilers – Week 3

Additional Examples

Ex2: “Brazil” language

- Basic datatype:
  * Integers:
    - Using hexadecimal signed values
    - Ex: LITERAL / CONSTANTS (2 bytes):

      $(0000)_{16}$ $(0)_{10}$ ... $(FFFF)_{16}$ $(65535)_{10}$.

    - *Remember: Hex: 0=0... , A=10, ... F=15*
- Variables (only using 16 Registers):
    - R0...RF (Registers: R_n, n=0...F)
    - Note: They use the prefix ‘@’ $(0040)_{16}$

Ex2: “Brazil” language

- Comments (single line):
    * Starts with # = $(0023)_{16}$
- Basic operations:
    * Input / Output:
        * SC (CodOp = 1): Scan: keyboard (to a register)
        * DP (CodOp = 2): Display: screen
    * Memory manipulation:
        * LD (CodOp = 3): Load (register / constant)
        * ST (CodOp = 4): Store (register)

Ex2: “Brazil” language

* Operations:
    * Arithmetic:
        * AD (CodOp = 5): Integer Addition
        * SU (CodOp = 6): Integer Subtration
        * MU (CodOp = 7): Integer Multiplication
        * DI (CodOp = 8): Integer Division
    * Logical:
        * AN (CodOp = 9): And operator
        * OR (CodOp = A): Or operator
        * NT (CodOp = B): Not operator

Ex2: “Brazil” language

* Functions:
    * BF: (CodOp = C): Beginning of Function
    * EF: (CodOp = D): End Function with return value (0000 = NULL)
    * Syntax:
        * Names: using ‘_’ for function names: $(005F)_{16}$
        * Scope: Using “:” $(003A)_{16}$ and finishing with EF.
        * Template:

```
BF _FUNCNAME_ PARAM_LIST:
    #COMMANDS
EF REG
```

Ex2: “Brazil” language

* Extra commands (controls):
    * ST: (CodOp = 0): Start Operation (beginning of program)
    * LF: (CodOp = E): Load function (returning value to a specific reg)
        - Syntax: **LF** RET _FUNCNAME_ PARAM_LIST
    * HT: (CodOp = F): Halt (end of execution).

NOTE:
In this hypothetical language, functions have indexes to label their names (until 15). The main program (not named) uses the index 0.

Ex2: “Brazil” language

- Examples:

  * Attribution:

```
LD R0
ST 0001
```

Intermed. rep.:

```
3 @0
4 0001
```

Binary:

```
01101000
1000001
```

  * Functions samples (note the indexes):

```
BF _ADD_ R0 R1:
LD R0
AD R1
ST R2
EF R2
```

Intermed. rep.:

```
# Func Index
1: _ADD_
# Representation:
C 1 @0 @1
3 @0
5 @1
4 @2
D @2
```

Ex2: “Brazil” language

- Examples:

  * Functions samples:

```
BF _MULTBYTWO_ R1:
LD R1
MU 0002
ST R1
EF R1
```

  # Intermed. repres:

```
# Func Index
2: _MULTBYTWO_
# Representation:
C 2 @1
7 0002
5 @1
4 @1
D @1
```

Ex2: “Brazil” language

- More fictional examples:

* Hello World:

```
# Func 3
BF _HELLO_:
DP 48 # Letter H
DP 45 # Letter E
DP 4C # Letter L
DP 4C # Letter L
DP 4F # Letter O
EF 00
```

* Invoking function:

```
ST
LF 00 _HELLO_
HT
```

Ex2: “Brazil” language

- Invoking functions:

* Sphere Volume:

```
# Func 4
BF _VOLUME_ R0:
LD R0
MU R0
MU R0
MU 0004
MU 7AAB # 31400
DI 7530    # 30000
ST R0
EF R0
```

* Invoking:

```
ST
SC R2
LF R3 _VOLUME_ R2
DP R3
HT
```

Open questions…

Any questions, let me know…

Image URL: https://cdn-az.allevents.in/events3/banners/26b150363d5757da8578fa6a1481585a368f12d4247adfe95837e6ea6c5ab2af-rimg-w1200-h549-gmir.jpg?v=1569691155

Compilers – Week 3

Thank you for your attention!