

Formal Languages and Compiler Design

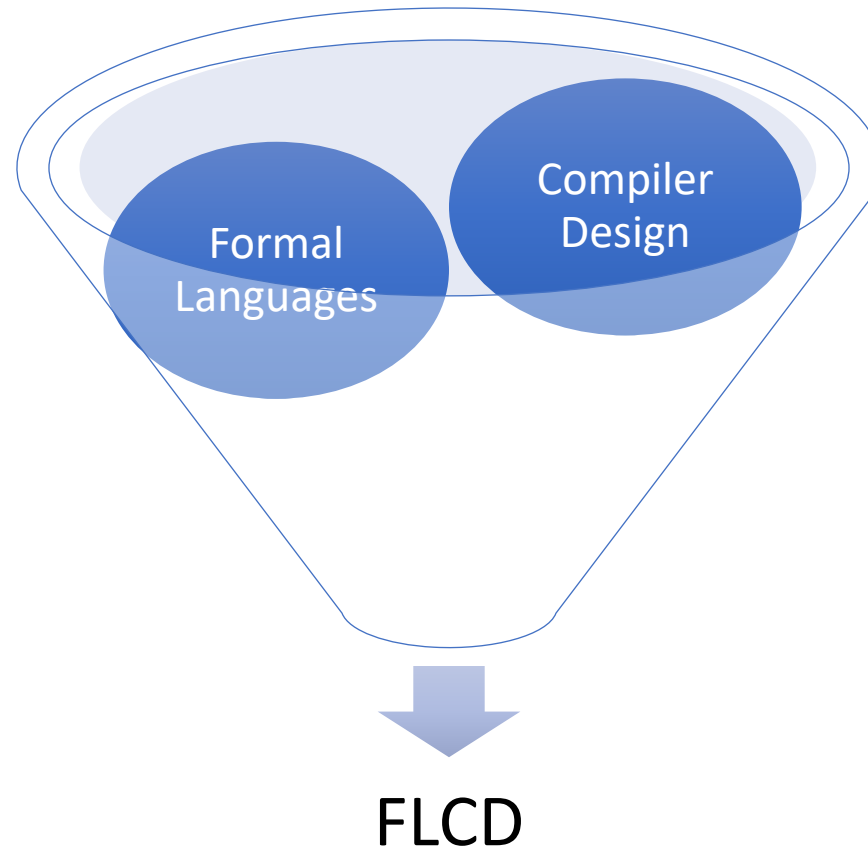
Símona Motogna

Why?

Historical reasons

Be a better programmer

Performant algorithms



Organization Issues

- Course – 2 h/ week
- Seminar – 2h/week
- Laboratory - 2 h/week

10 presences – seminar
12 presences - lab

PRESENCE IS MANDATORY

Most interesting stuff for students

- Organization.pptx

References

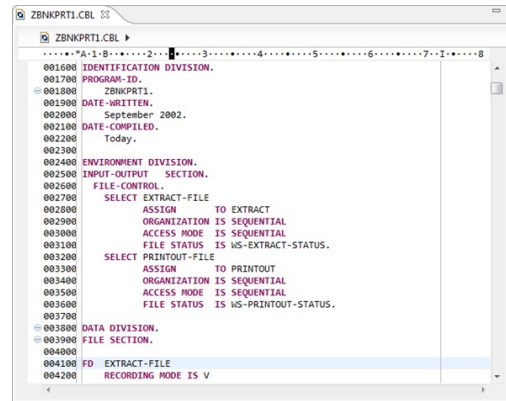
- See [fișa disciplinei](#)

```
import time
```

```
def count(limit):
    result = 0
    for a in range(1, limit + 1):
        for b in range(a + 1, limit + 1):
            for c in range(b + 1, limit + 1):
                if c * c > a * a + b * b:
                    break

                if c * c == (a * a + b * b):
                    result += 1

    return result
```



```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>
```

```
struct stats { int count; int sum; int sum_squares; };
```

```
void stats_update(struct stats * s, int x, bool reset) {
    if (s == NULL) return;
    if (reset) * s = (struct stats) { 0, 0, 0 };
    s->count += 1;
    s->sum += x;
    s->sum_squares += x * x;
}
```

```
double mean(int data[], size_t len) {
    struct stats s;
    for (int i = 0; i < len; ++i)
        stats_update(&s, data[i], i == 0);
    return ((double)s.sum) / ((double)s.count);
}
```

```
void main() {
    int data[] = { 1, 2, 3, 4, 5, 6 };
    printf("MEAN = %lf\n", mean(data, sizeof(data) / sizeof(data[0])));
}
```

```
package rentalStore;
import java.util.Enumeration;
import java.util.Vector;

class Customer {
    private String _name;
    private Vector<Rental> _rentals = new Vector<Rental>();

    public Customer(String name) {
        _name = name;
    }

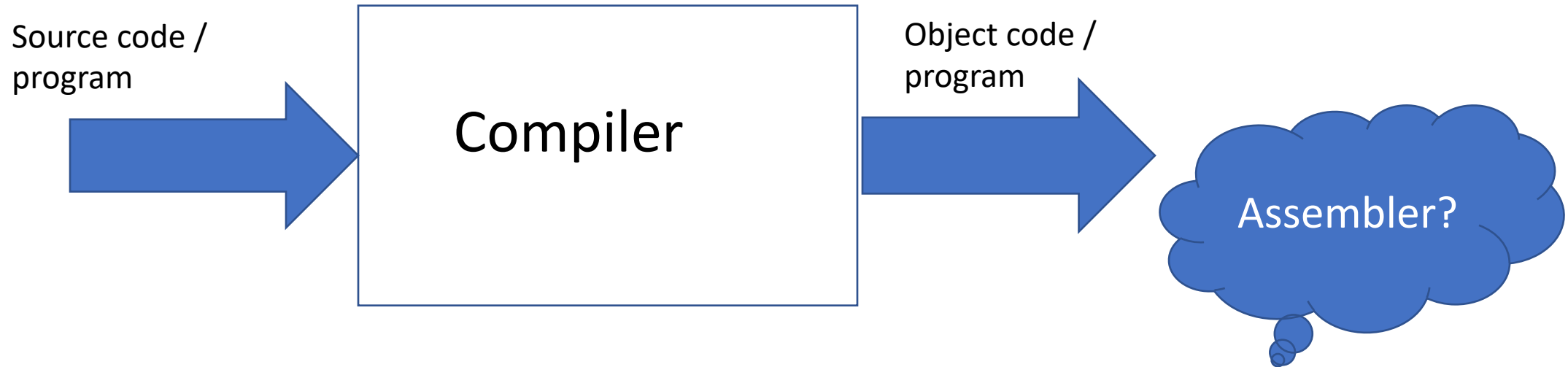
    public String getMovie(Movie movie) {
        Rental rental = new Rental(new Movie("", Movie.NEW_RELEASE), 10);
        Movie m = rental._movie;
        return movie.getTitle();
    }

    public void addRental(Rental arg) {
        _rentals.addElement(arg);
    }

    public String getName() {
        return _name;
    }
}
```

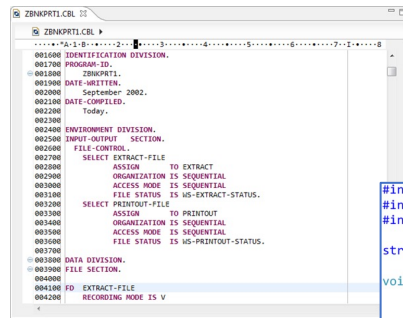
```
190      C
191          PIN=0.02
192          IF (DDT.NE.0.0) THEN
193              DT=DDT
194          ELSE
195              DT=PIN
196          ENDIF
197          WRITE(*,'(A)') '    PLEASE ENTER NAME OF OUTPUT FILE (FOR EXAMPLE
198          * B:ZZ.DAT)'
199          READ(*,'(A)') FNAMEO
200          OPEN(6,FILE=FNAMEO,STATUS='UNKNOWN')
201          PV=HFLX/TH
202          RS=NEQ*ROU*KD/TH
203          C0=CS
204      C
205          TIME=0.0D0
206          EF=0.0D0
207          5  CONTINUE
208              GAMMA=DT/(2.D0*DX*DX)
209              BETA=DT/DX
210              IF ((BETA*PV).GT.0.50D0) GO TO 7
211              IF ((GAMMA*D/(BETA*PV)).LT.0.50D0) GO TO 6
212              GO TO 8
213          6  DX=DX/2
214              GO TO 5
215          7  DT=DT/2
216              GO TO 5
217          8  CONTINUE
218              N=COL/DX
219              NM1=N-1
220              NM2=N-2
221              NP1=N+1
222              GAMMA=DT/(2*DX*DX)
```

What is a compiler?



```
import time

def count(limit):
    result = 0
    for a in range(1, limit + 1):
        for b in range(a + 1, limit + 1):
            for c in range(b + 1, limit + 1):
                if c * c > a * a + b * b:
                    break
                if c * c == (a * a + b * b):
                    result += 1
    return result
```



```
#include <stdlib.h>
#include <stdio.h>
#include <stdbool.h>

struct stats { int count; int sum; int sum_squares; };

void stats_update(struct stats * s, int x, bool reset) {
    if (s == NULL) return;
    if (reset) * s = (struct stats) { 0, 0, 0 };
    s->count += 1;
    s->sum += x;
    s->sum_squares += x * x;
}

double mean(int data[], size_t len) {
    struct stats s;
    for (int i = 0; i < len; ++i)
        stats_update(&s, data[i], i == 0);
    return ((double)s.sum) / ((double)s.count);
}

void main() {
    int data[] = { 1, 2, 3, 4, 5, 6 };
    printf("MEAN = %f\n", mean(data, sizeof(data) / sizeof(data[0])));
}
```

```
package rentalStore;
import java.util.Enumeration;
import java.util.Vector;

class Customer {
    private String _name;
    private Vector<Rental> _rentals = new Vector<Rental>();

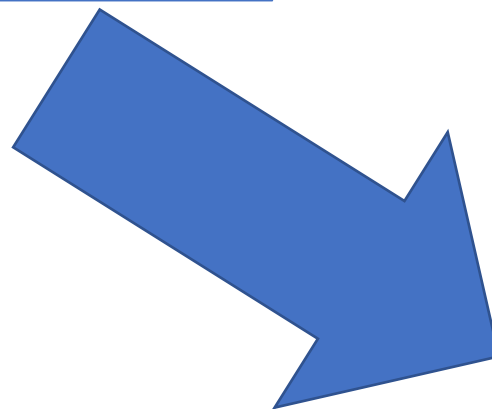
    public Customer(String name) {
        _name = name;
    }

    public String getMovie(Movie movie) {
        Rental rental = new Rental(new Movie("", Movie.NEW_RELEASE), 10);
        Movie m = rental._movie;
        return movie.getTitle();
    }

    public void addRental(Rental arg) {
        _rentals.addElement(arg);
    }

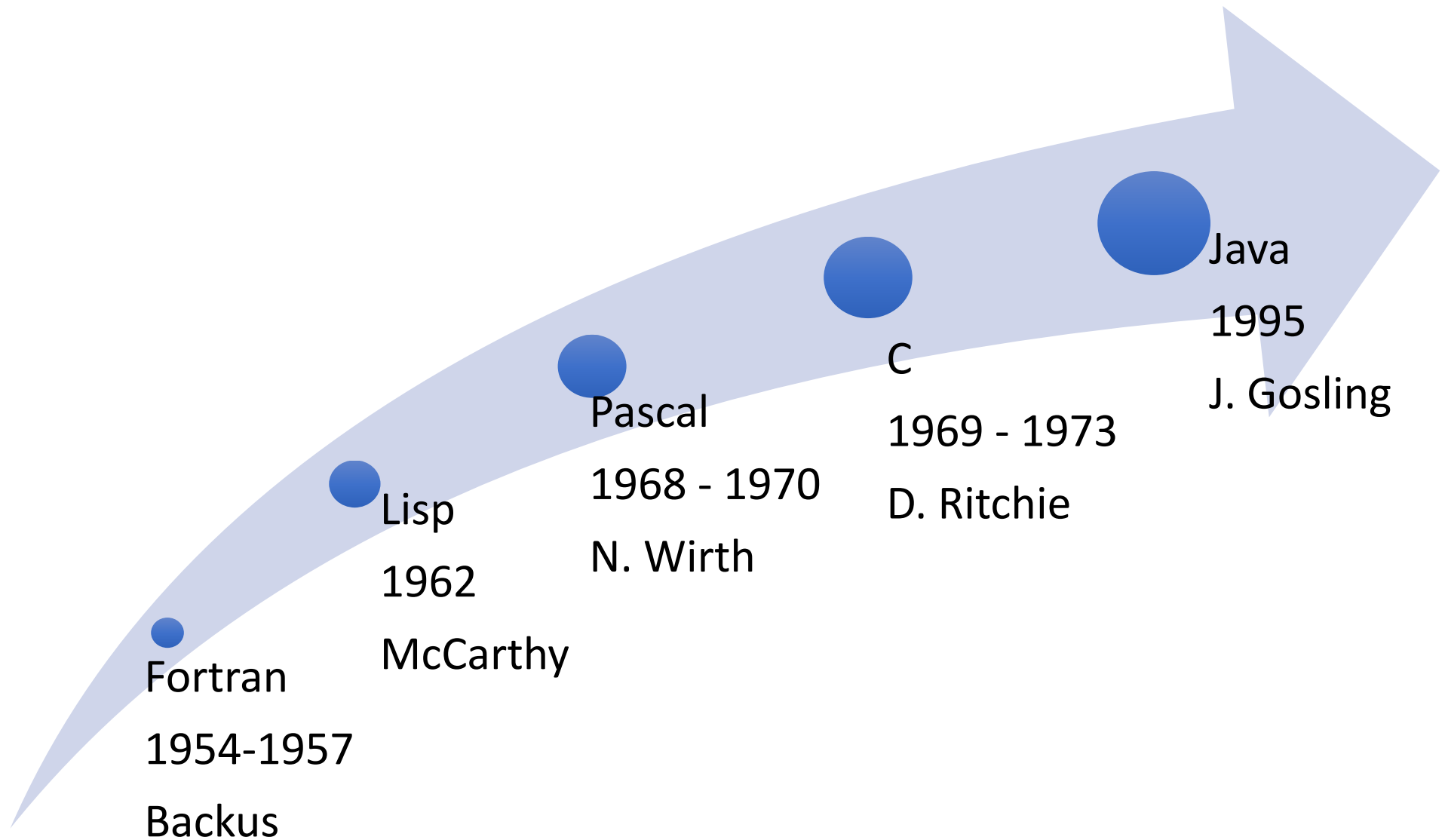
    public String getName() {
        return _name;
    }
}
```

```
190      C      PIN=0.02
191
192      IF (DOT.NE.0.0) THEN
193          DT=DOT
194      ELSE
195          DT=PIN
196      ENDIF
197      WRITE(*, '(A)') ' PLEASE ENTER NAME OF OUTPUT FILE (FOR EXAMPLE
198      * B:Z1.DAT)'
199      READ(*, '(A)') FNAMEO
200      OPEN (6, FILE=FNAMEO, STATUS='UNKNOWN')
201      PV=WM/LX/TR
202      B=REC*MOD/KD/TR
203      CO=CS
204
205      C
206      TIME=0.000
207      EF=0.000
208      5      CONTINUE
209      GAMMA=DT/(2.00*DX*DX)
210      IF ((BETA*PV).GT.0.5000) GO TO 7
211      IF ((GAMMA*D)/(BETA*PV)).LT.0.500) GO TO 6
212      GO TO 8
213      6      DX=DX/2
214      GO TO 5
215      7      DT=DT/2
216      GO TO 5
217      8      CONTINUE
218      N=COL/DX
219      NN1=N-1
220      NN2=N-2
221      NN1=N+1
222      GAMMA=DT/(2*DX*DX)
```

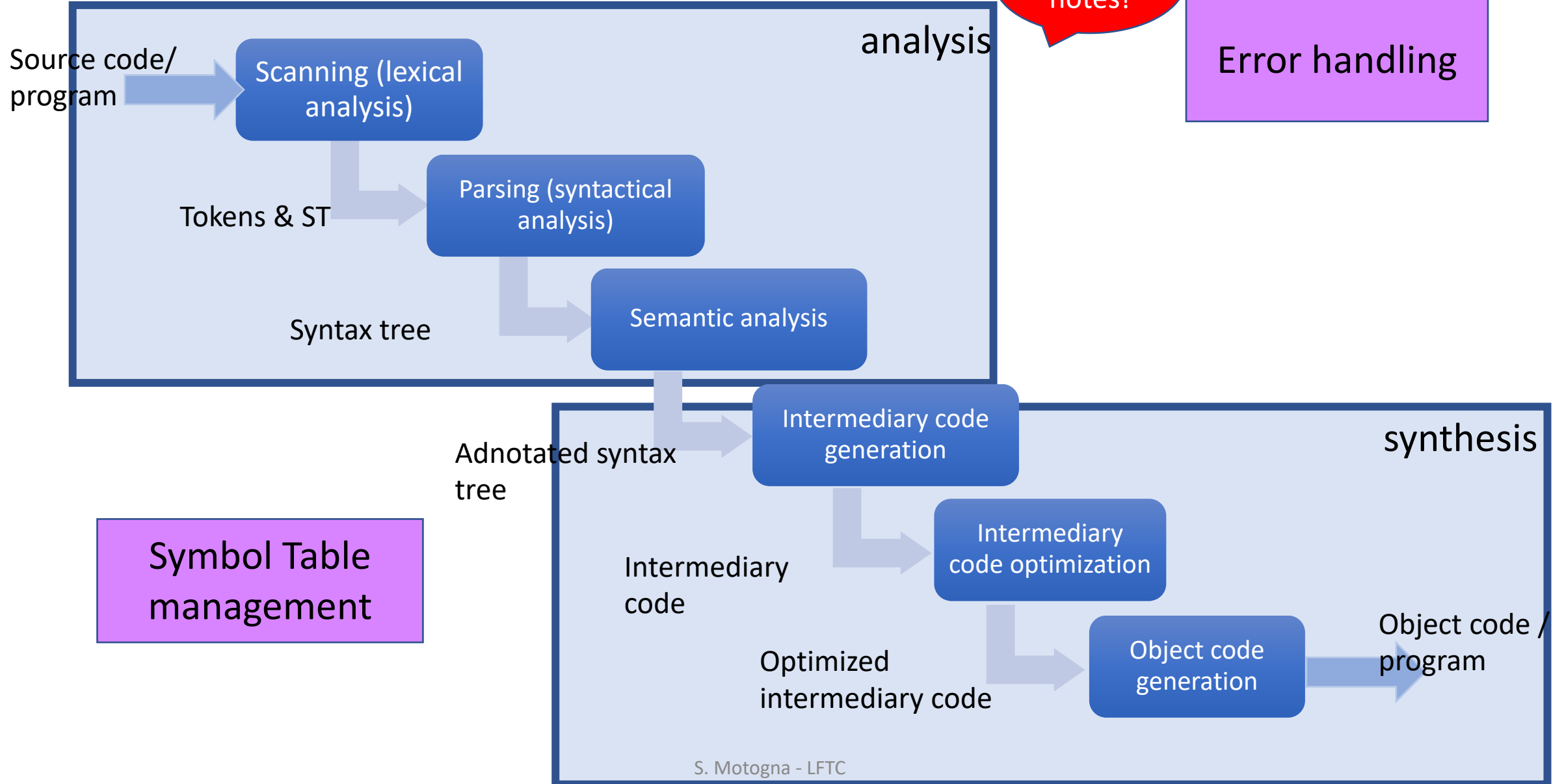


```
00000000 0000 0001 0001 1010 0010 0001 0004 0128
00000010 0000 0016 0000 0028 0000 0010 0000 0020
00000020 0000 0001 0004 0000 0000 0000 0000 0000
00000030 0000 0000 0000 0010 0000 0000 0000 0204
00000040 0004 8384 0084 c7c8 00c8 4748 0048 e8e9
00000050 00e9 6a69 0069 a8a9 00a9 2828 0028 fdfc
00000060 00fc 1819 0019 9898 0098 d9d8 00d8 5857
00000070 0057 7b7a 007a bab9 00b9 3a3c 003c 8888
00000080 8888 8888 8888 8888 288e be88 8888 8888
00000090 3b83 5788 8888 8888 7667 778e 8828 8888
000000a0 d61f 7abd 8818 8888 467c 585f 8814 8188
000000b0 8b06 e8f7 88aa 8388 8b3b 88f3 88bd e988
000000c0 8a18 880c e841 c988 b328 6871 688e 958b
000000d0 a948 5862 5884 7e81 3788 1ab4 5a84 3eec
000000e0 3d86 dc88 5cbb 8888 8888 8888 8888 8888
000000f0 8888 8888 8888 8888 8888 8888 8888 0000
00001000 0000 0000 0000 0000 0000 0000 0000 0000
*
00001030 0000 0000 0000 0000 0000 0000 0000 0000
0000103e
```


A little bit of history ...



Structure of a compiler



Chapter 1. Scanning

Definition = treats the source program as a sequence of characters, detect lexical tokens, classify and codify them

INPUT: source program

OUTPUT: PIF + ST

Algorithm Scanning v1

```
While (not.eof) do  
    detect(token);  
    classify(token);  
    codify(token);  
End_while
```

Detect

Take
notes!

I am a student. I am
Simona

- Separators => **Remark 1)**

if (x==y) {x=y+2}

- Look-ahead => **Remark 2)**

Classify

- Classes of tokens:
 - Identifiers
 - Constants
 - Reserved words (keywords)
 - Separators
 - Operators
- If a token can NOT be classified => LEXICAL ERROR

Codify

- May be codification table

OR

code for identifiers and constants

- Identifier, constant => Symbol Table (ST)
- PIF = Program Internal Form = array of pairs
- pairs (token, position in ST)



identifier, constant

Algorithm Scanning v2

```
While (not.eof) do  
    detect(token);  
    if token is reserved word OR operator OR separator  
        then genPIF(token, 0)  
    else  
        if token is identifier OR constant  
            then index = pos(token, ST);  
                genPIF(token, index)  
            else message "Lexical error"  
        endif  
    endif  
endwhile
```

a=a+b

FIP

(id,1)

(=,0)

(id,1)

(+,0)

(id,2)

ST

1 a

2 b

Remarks:

- genPIF = adds a pair (token, position) to PIF
- Pos(token,ST) – searches *token* in symbol table *ST*; if found then return position; if not found insert in SR and return position
- Order of classification (reserved word, then identifier)
- If-then-else imbricate => detect error if a token cannot be classified