2020 SASA Internship
Lecture 2. How to Create a Programming Language

School File Managing Language

2105 Sang kwon Park
2208 Hyungsuk Song

CONTENTS

Contents 1
What We Learned

Programming Language Classification
- Imperative / Declarative Language
- Logic / Expression La

Parsing
- Internal DSL VS external DSL

Project

ranslation

Project → Making our own DSL!!

Langua
- Interp
- Overv

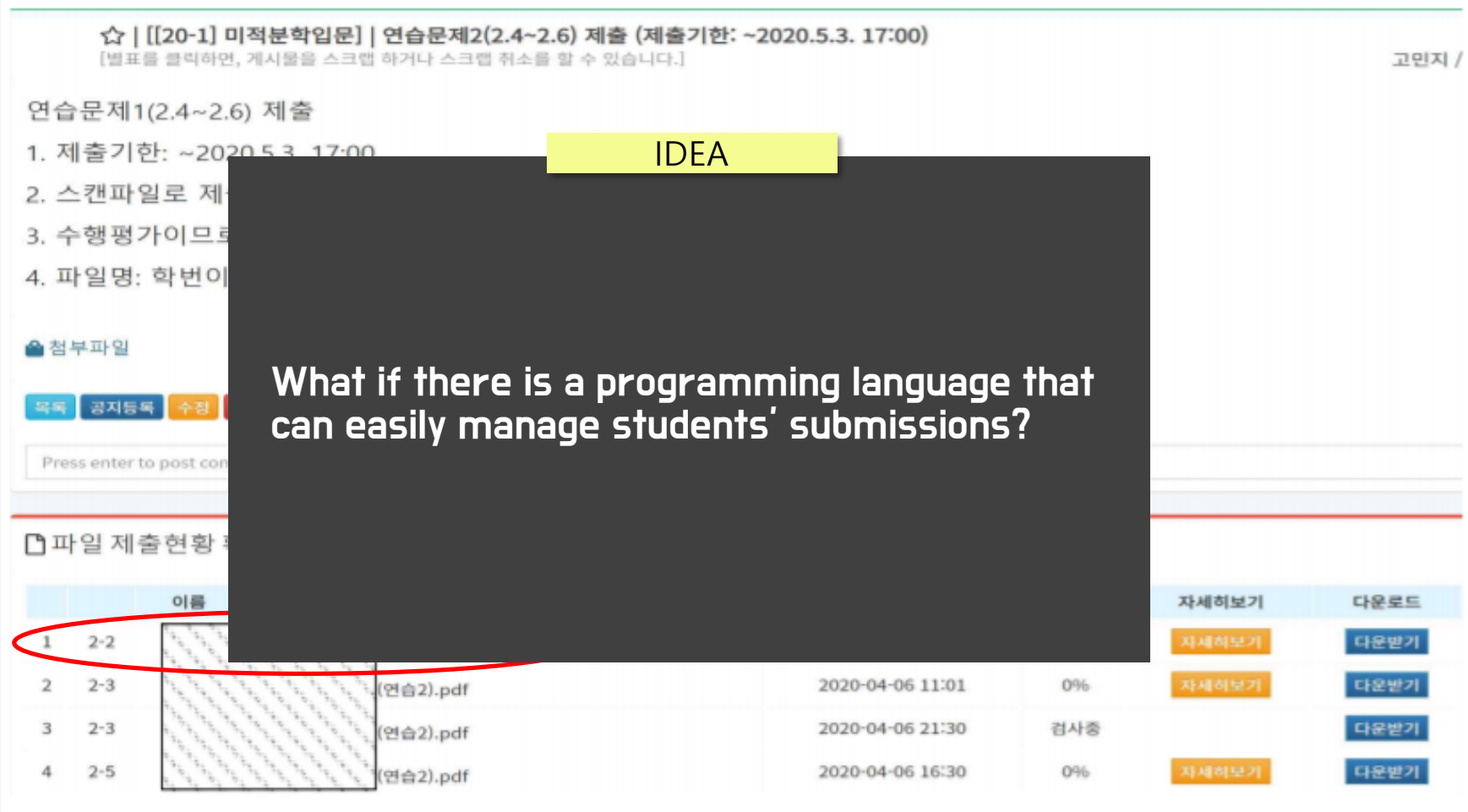
Lexical
Domain-Specific Language

ermediate Code generation

Contents 2
Selecting Domain

2208송형석_4단원연습문제.pdf	2020-05-17 오후 9:10	Microsoft Edge P...	931KB
2208송형석_8장연습문제.pdf	2020-05-31 오후 6:53	Microsoft Edge P...	1,506KB
2208송형석_13장연습문제.pdf	2020-06-14 오후 11:16	Microsoft Edge P...	629KB
2208송형석_14장사전검사지.jpg	2020-05-30 오후 11:11	JPG 파일	436KB
2208송형석_16연습문제.pdf	2020-07-19 오후 10:49	Microsoft Edge P...	554KB
2208송형석_Chaper2_연습문제..pdf	2020-05-03 오전 12:28	Microsoft Edge P...	1,082KB
2208송형석_Chapter3_연습문제.pdf	2020-05-03 오후 8:14	Microsoft Edge P...	438KB
2208송형석_chapter4_사전검사지.pdf	2020-05-01 오후 9:52	Microsoft Edge P...	418KB
2208송형석_chapter7_사전검사지.pdf	2020-05-01 오후 9:52	Microsoft Edge P...	522KB
Chapter2_사후검사지_2208송형석.pdf	2020-05-01 오후 4:37	Microsoft Edge P...	2,032KB
Motion in two dimensions_2208송형석....	2020-05-17 오후 5:20	Microsoft Edge P...	1,490KB
Relative motion_2208송형석.pdf	2020-05-17 오후 5:21	Microsoft Edge P...	1,790KB
사후검사지_상대운동_2208송형석.pdf	2020-05-17 오후 9:10	Microsoft Edge P...	1,659KB
창의적산출물과제_2208송형석.hwp	2020-07-14 오후 10:08	한컴오피스 2018 ...	264KB

Contents 2
Selecting Domain



IDEA

What if there is a programming language that can easily manage students' submissions?

출처 : 고민지 선생님 미적연습OT 강의자료

Contents 3
Selecting Operations

- Rename a file / Rename all files of form that we want.
- Print current directory path where user is located
- Create a directory with the desired name
- Create a class list
- Adding students to the class list
- Go to desired directory
- Print directories and files in current directory
- Collect the files for the specified class

Contents 3
Creating Syntax

- rename NAME to NAME
- whereami
- make dir NAME
- make class NAME
- insert NUMBER NAME to class NAME
- move to NAME
- list | list DIR
- list_file | list_file DIR
- for LPAREN LIST RPAREN in LPAREN DIR RPAREN LBRACKET rename to NAME RBRACKET

  ex) for (class A) in (list_file) { rename to 2208_HyungsukSong.docx }
- for LPAREN LIST RPAREN move to NAME

  ex) for (class A) move to CLASS_A

Contents 3

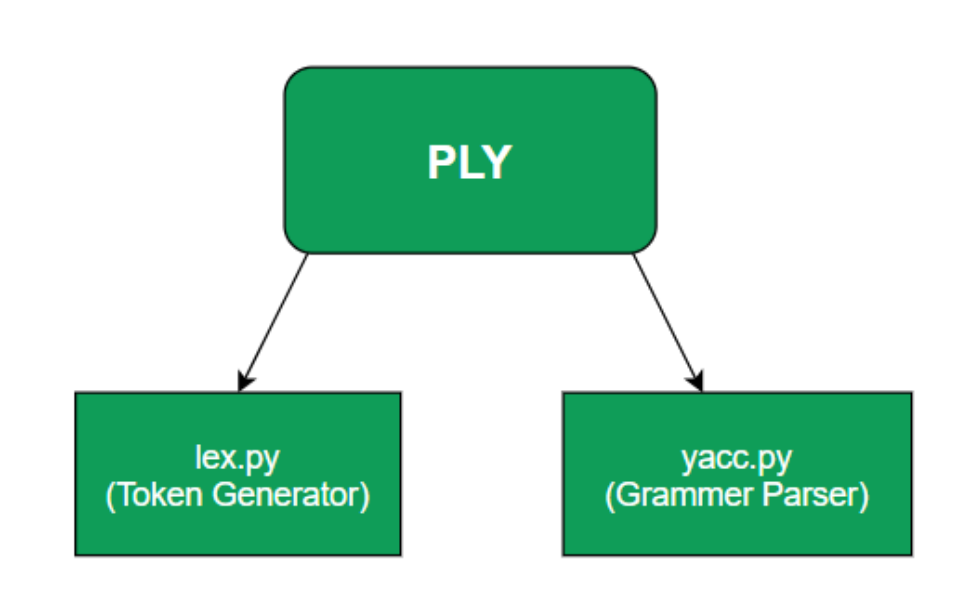
Creating Syntax

```
Stmt : FILE
     | DIR
     | LIST

DIR : make dir NAME
    | whereami
    | move to NAME

FILE : rename NAME to NAME

LIST : list_file
     | list_file DIR
     | list
     | list DIR
     | make class NAME
     | Class NAME
     | insert NUMBER NAME to class NAME
     | insert NAME NUMBER to class NAME
     | for LPAREN LIST RPAREN in LPAREN DIR RPAREN LBRACKET rename to
                                                          NAME RBRACKET
     | for LPAREN LIST RPAREN move to NAME
```

Contents 4
Implementation

With ply(Python Lex-yacc)

PLY ? →
implemention of lex
and yacc parsing tools
for python



Contents 4
Implementation

```python
def stmt(p):
    '''type :
        | FILE
        | DIR
        | LIST'''
    p[0] = p[1]
```

```python
tokens = [
    'NAME', 'NUMBER',
    'LPAREN', 'RPAREN',
    'LBRACKET', 'RBRACKET'
]

reserved = {
    'rename': 'rename', 'whereami': 'whereami',
    'make': 'make', 'to': 'to',
    'for': 'for', 'dir': 'dir',
    'in': 'in', 'class': 'class',
    'move': 'move', 'get_back': 'get_back',
    'list': 'list', 'list_file': 'list_file',
    'insert': 'insert',
}
tokens += reserved.values()

t_ignore = ' \t'
t_LPAREN = '\('
t_RPAREN = '\)'
t_LBRACKET = '\{'
t_RBRACKET = '\}'
```

```python
def t_NAME(t):
    r'[a-zA-Z_\.\%][a-zA-Z0-9_\.\%]*'
    if t.value in reserved:
        t.type = reserved[t.value]
    return t


def t_NUMBER(t):
    r'[0-9]+'
    t.value = int(t.value)
    return t
```

Contents 4
Implementation

```python
def p_rename(p):
    '''FILE : rename NAME to NAME'''
    cur_file = os.path.join(os.getcwd(), p[2])
    new_file = os.path.join(os.getcwd(), p[4])
    os.rename(cur_file, new_file)
    p[0] = new_file
```

```python
def p_MakeDir(p):
    'DIR : make dir NAME'
    if not os.path.exists(os.path.join(os.getcwd(), p[3])):
        os.mkdir(os.path.join(os.getcwd(), p[3]))
    p[0] = os.path.join(os.getcwd(), p[3])


def p_curDir(p):
    'DIR : whereami'
    p[0] = os.getcwd()
```

```python
def p_listDir(p):
    '''LIST : list
    | list DIR'''
    # print(os.listdir(os.getcwd()))
    cur = os.getcwd()
    if len(p) > 2:
        cur = p[2]
    target = os.listdir(cur)
    # print("----------------")
    rem = search(cur)
    rem = [os.path.split(i)[-1] for i in rem]
    target = [i for i in target if not i in rem]
    p[0] = target


def p_listFile(p):
    '''LIST : list_file
    | list_file DIR'''
    cur = os.getcwd()
    if len(p) > 2:
        cur = p[2]
    target = os.listdir(cur)
    p[0] = target
```

Contents 4
Implementation

```python
def p_moveDir(p):
    'DIR : move to NAME'
    if os.path.exists(os.path.join(os.getcwd(), p[3])):
        os.chdir(os.path.join(os.getcwd(), p[3]))
        p[0] = p[3]


def p_makeClass(p):
    'LIST : make class NAME'
    C[p[3]] = {}
    p[0] = C[p[3]]
```

```python
def p_insertStu(p):
    'LIST : insert NUMBER NAME to class NAME'
    C[p[6]][p[2]] = p[3]
    p[0] = C[p[6]]
```

Contents 4
Implementation

```
def p_for(p):
    '''LIST : for LPAREN LIST RPAREN in LPAREN DIR RPAREN LBRACKET rename to NAME RBRACKET
    | for LPAREN LIST RPAREN move to NAME'''
    if len(p) < 10:
        tmp = {}
        for file in os.listdir(os.getcwd()):
            for key, value in p[3].items():
                print(file, key, value, file.find(str(key)))
                if file.find(str(key)) >= 0 or file.find(value) >= 0:
                    cur_file = os.path.join(os.getcwd(), file)
                    t = os.path.join(os.getcwd(), p[7])
                    new_file = os.path.join(t, file)
                    tmp[cur_file] = new_file
        for cur_file, new_file in tmp.items():
            if os.path.exists(cur_file):
                print("rename : \n" + cur_file + "\nto\n" + new_file)
                os.rename(cur_file, new_file)
        p[0] = tmp
    else:
```

Contents 4
Implementation

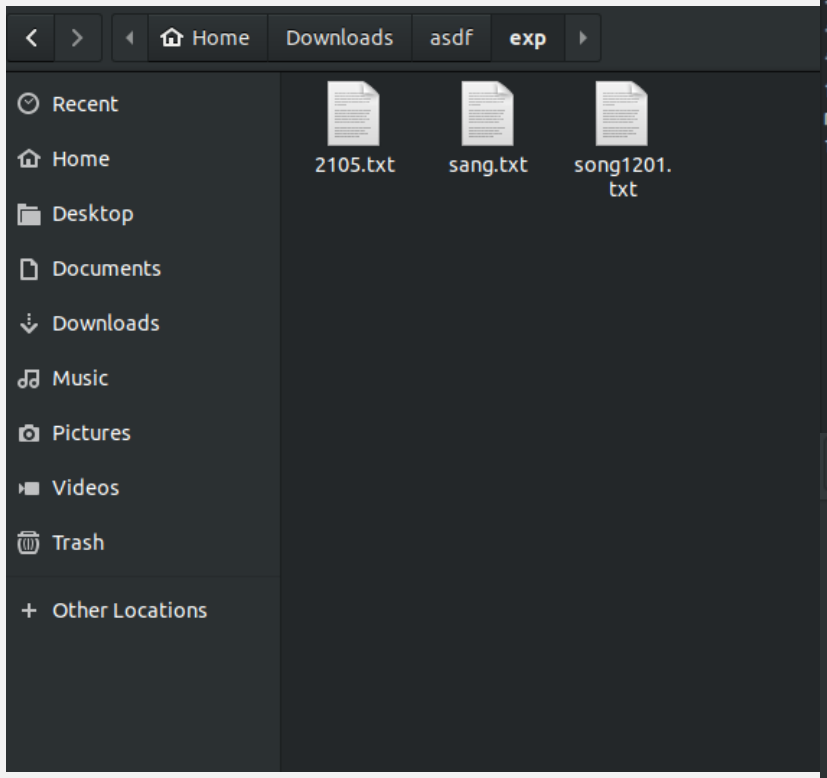
```
else:
    tmp = {}
    for file in p[7]:
        for key, value in p[3].items():
            if file.find(str(key)) >= 0 or file.find(value) >= 0:
                cur_file = os.path.join(os.getcwd(), file)
                t = p[12].replace("%name", value)
                t = t.replace("%num", str(key))
                new_file = os.path.join(os.getcwd(), t)
                tmp[cur_file] = new_file
    for cur_file, new_file in tmp.items():
        if os.path.exists(cur_file):
            print("now rename : \n" + cur_file + "\nto\n" + new_file)
            os.rename(cur_file, new_file)
```

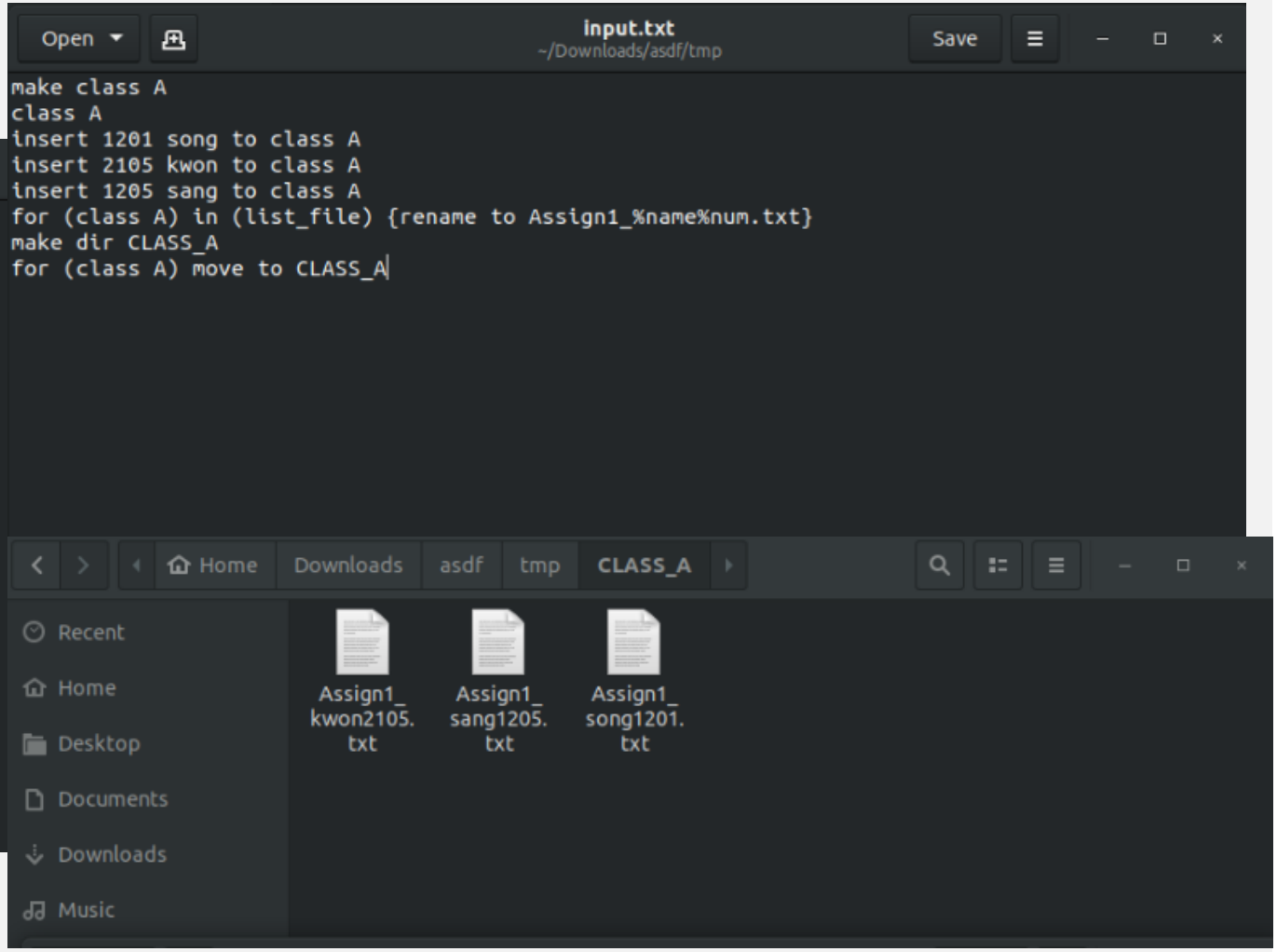
Contents 4
Implementation

```python
while True:
    print("(FileManager) ", end='')
    tmp = input()
    while tmp[-1] != ';':
        tmp = tmp + input()
    tmp = tmp[:-1]
    if tmp == "exit":
        print("Terminating...")
        break
    res = parser.parse(tmp)  # the input
    lex.input(tmp)
    # print(lexer.token())
    print(res)
```

```python
file = open("input.txt", 'r')
line = file.readline()
parser = yacc.yacc()

while line:
    tmp = line

    res = parser.parse(tmp)  # the input
    lex.input(tmp)
    print(res)
    line = file.readline()
```

Contents 4
Execution





Contents 5

Self Feedback

Thank you