Chapter 8-1

Text Processing

Contents

- String Basic Operations
- More String Operations
- Case Study: Processing e-Mail Addresses
- Case Study: Processing Dates
- Computational Problem

String Basic Operations

- 문자열 기본 연산
  - Python에서는 문자열에 대한 다양한 연산을 지원 함
  - 문자열의 문자를 인덱스로 접근(Indexing)하거나 문자열의 길이(Length)를 구하거나, 문자열의 덧셈(Concatenation) 및 자르기(Substring) 등이 가능 함

```
x = 'pop'
y = 'corn'
```

Operation	Example	Result
Indexing	x[1]	'o'
Length	len(x)	3
Concatenation	x + y	'popcorn'
Substring	x[1:3]	'op'
Searching	x.index('o')	1

String Basic Operations

Exercise!

```
>>> x = 'pop'
>>> y = 'corn'
>>> z = x + y

>>> z

>>> z[0]

>>> len(z)

>>> z[2:4]

>>> z.index('c')
```

```
>>> addr = 'stu@yonsei.ac.kr'

>>> aIndex = addr.index('@')

>>> length = len(addr)

>>> username = addr[0:aIndex]

>>> hostname = addr[aIndex+1:
                    length]
```

More String Operations

- 자주 사용되는 문자열 연산
  - 문자열 순방향 및 역방향 검색(find 및 rfind), 문자열 치환(replace), 문자열 앞뒤의 특정문자 제거(strip) 및 문자열 분할(split), 알파벳 혹은 숫자로 구성되었는지 확인, 대문자 및 소문자 변환 등이 가능

```
x = 'pop', y = ' corn ', z = 'A+B+C'
```

Operations	Example	Result
find(s)	x.find('p')	0
rfind(s)	x.rfind('p')	2
replace(s, t)	x.replace('p', 'm')	'mom'
strip(s)	y.strip(' ')	'corn'
split(s)	z.split('+')	['A','B','C']
isalpha()	x.isalpha()	True
isdigit()	x.isdigit()	False
lower()	x.lower()	'pop'
upper()	x.upper()	'POP'

More String Operations

Exercise!

아래 date를 replace() 함수를 사용하여 '2017-07-31'로 수정하세요

```
date = '2017/07/31'
```

아래 url에 split() 함수를 사용하여 영문 단어들을 추출하여 리스트로 저장하세요

```
url = 'www.yonsei.ac.kr'
```

사용자로 부터 영문자 한 개를 입력받아 answer 변수에 대문자로 저장하는 명령문을 작성하세요

Case Study: Processing e-Mail Addresses

- e-mail 주소의 분석

e-Mail Analysis for Any Valid e-Mail

address ← *readAddressFromUser()*
ampersatIndex ← address.indexOf("@")
length ← address.length
username ← address.substring(0, ampersatIndex)
hostsite ← address.substring(ampersatIndex+1, 28)
extension ← address.substring(length-3, length)

```
address = readAddressFromUser()
ampersatIndex = address.index('@')
length = len(address)
username = address[0:ampersatIndex]
hostsite = address[ampersatIndex+1:length]
extension = address[address.rfind('.')+1:length]
```

Case Study: Processing Dates

- 날짜 형식의 변환: 유럽식 → 미국식

e-Mail Analysis for the Beatles

edate ← *readDateFromUser()*
day ← edate.substring(0, 2)
month ← edate.substring(3, 5)
year ← edate.substring(6, 10)
adate ← month + "/" + day + "/" + year

```
edate = readDateFromUser()
day = edate[0:2]
month = edate[3:5]
year = edate[6:10]
adate= month + '/' + day + '/' + year
```

혹은

```
edate = readDateFromUser()
day, month, year = edate.split('/')
adate= month + '/' + day + '/' + year
```

Computational Problem

The Problem

주민등록번호 분석 프로그램

사용자로부터 주민등록번호를 '000000-0000000'와 같은 형식으로 입력 받아 아래 정보를 출력합니다.

- 생년월일: 'YYYY-MM-DD' 형식으로 출력
- 성별: 'Male' 혹은 'Female'로 출력

주민등록번호 분석 프로그램

Problem Analysis

문제의 요구사항

- 사용자가 주민등록번호를 입력하면, 올바른 형식인지 확인하여야 함
- 올바른 형식의 주민번호를 입력 받으면, 생년월일과 성별을 주민등록번호로 부터 추출하고 이를 가공하여 출력하여야 함

주민등록번호의 구성

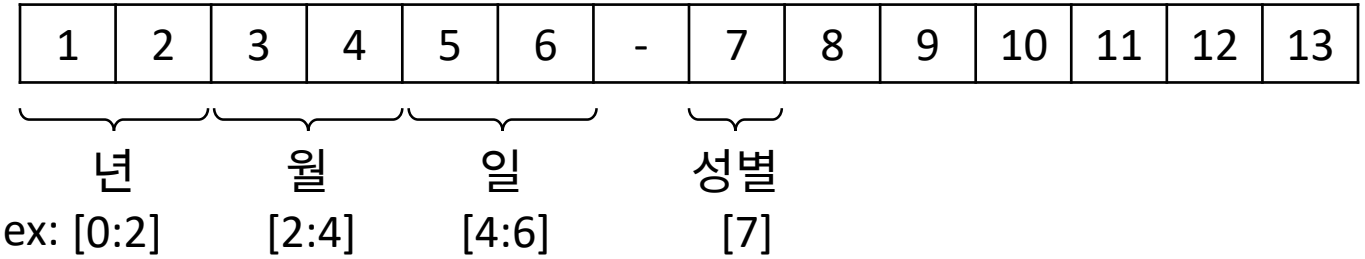
주민등록번호 분석 프로그램

Problem Analysis

문제의 요구사항

- 사용자가 주민등록번호를 입력하면, 올바른 형식인지 확인하여야 함
- 올바른 형식의 주민번호를 입력 받으면, 생년월일과 성별을 주민등록번호로 부터 추출하고 이를 가공하여 출력하여야 함

주민등록번호의 구성



주민등록번호 분석 프로그램

Data Representation

주민등록번호문자열

rrn

(Resident Registration Number)

주민등록번호 분석 프로그램

Algorithmic Thinking

올바른 주민등록번호의 조건

① 문자열에 '-' 문자가 인덱스 6에 위치해야 함

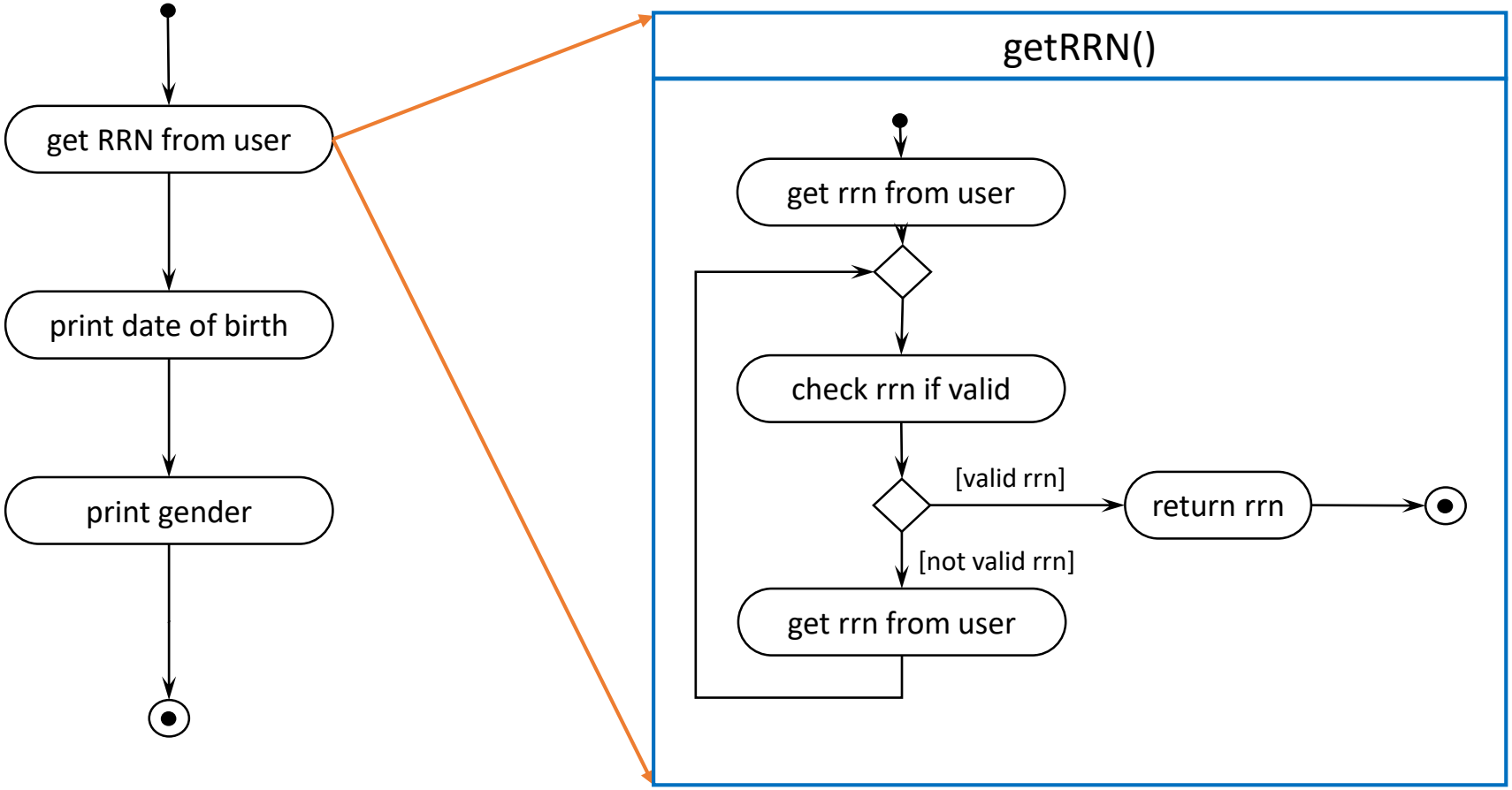
② 인덱스 0부터 5까지의 문자열은 모두 숫자이어야 함

③ 인덱스 7부터 13까지의 문자열은 모두 숫자이어야 함

④ 문자열의 문자의 개수는 14개이어야 함

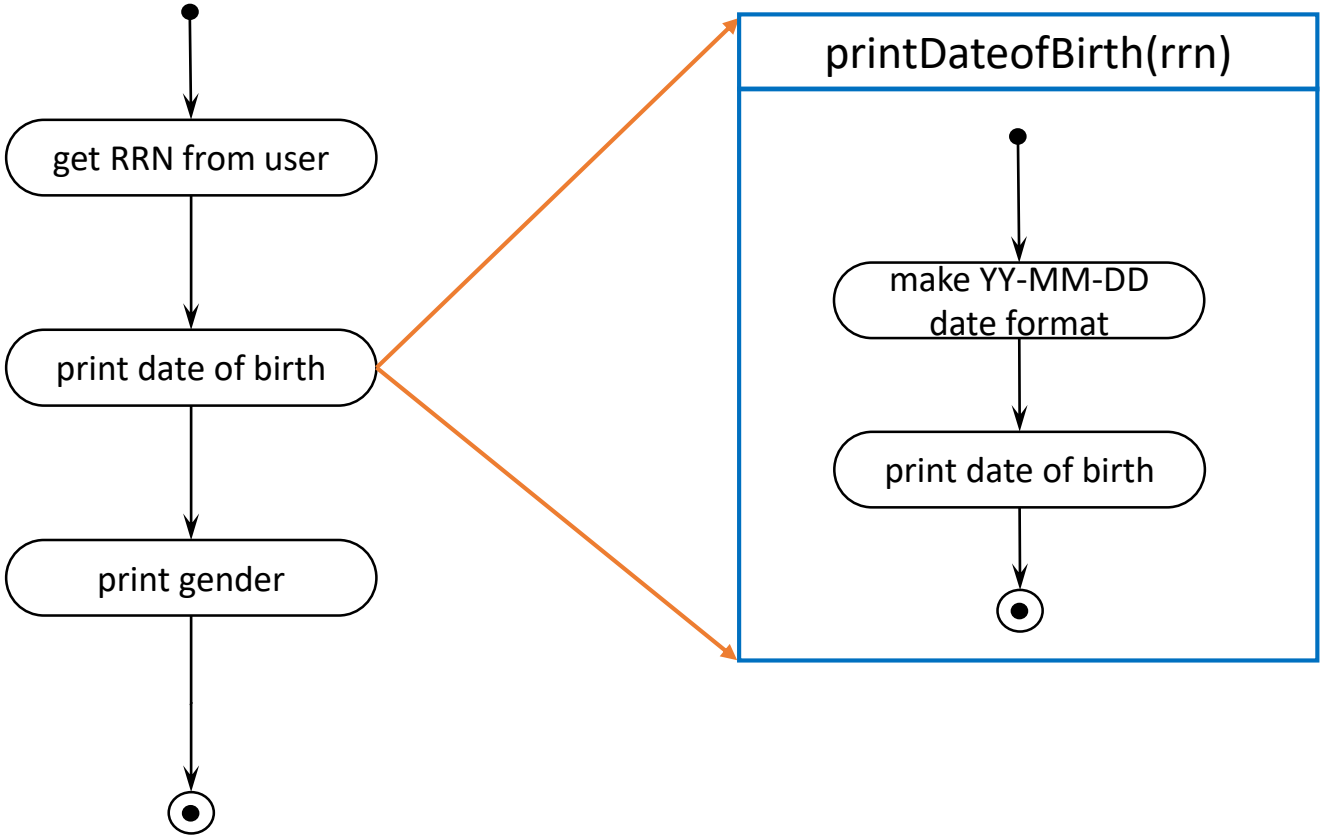
주민등록번호 분석 프로그램

Algorithmic Thinking & Decomposition



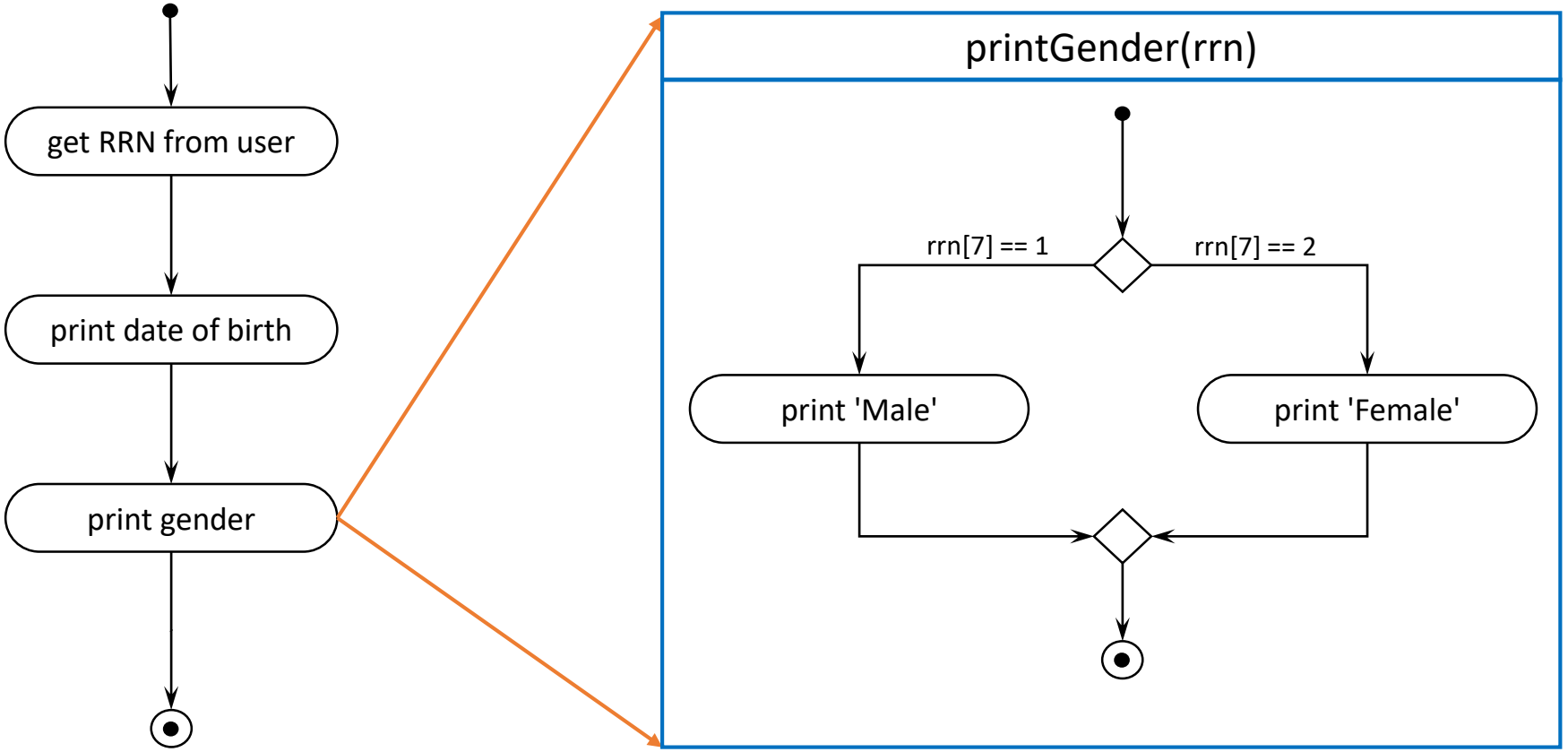
주민등록번호 분석 프로그램

Algorithmic Thinking & Decomposition



주민등록번호 분석 프로그램

Algorithmic Thinking & Decomposition



주민등록번호 분석 프로그램

Program Design

Program Introduction

Displaying welcome messages

Input

input rrn from user

Processing

format date of birth and determine gender

Display results

Display date of birth and gender

주민등록번호 분석 프로그램

Program Implementation

```
def getRRN():
    rrn = input('Enter an RRN: ')
    while not validRRN(rrn):
        print('Wrong format')
        rrn = input('Enter an RRN: ')
    return rrn

def validRRN(rrn):
    dashIndex = rrn.find('-')
    valid = (dashIndex == 6) and \
            rrn[0:dashIndex].isdigit() and \
            rrn[dashIndex+1:].isdigit() and \
            len(rrn) == 14
    return valid
```

주민등록번호 분석 프로그램

Program Implementation

```
def printDateOfBirth(rrn):
    dateOfBirth = rrn[0:2] + '-' + rrn[2:4] + '-' + rrn[4:6]
    print('Date of Birth:',dateOfBirth)

def printGender(rrn):
    if rrn[7] == '1':
        print('Gender: Male')
    else:
        print('Gender: Female')
```

주민등록번호 분석 프로그램

Program Implementation

```
rrn = getRRN()
printDateOfBirth(rrn)
printGender(rrn)
```

주민등록번호 분석 프로그램

Program Execution

```
Enter an RRN: 850123-1234
Wrong format
Enter an RRN: 850123-1234567
Date of Birth: 85-01-23
Gender: Male
>>>
```

*참고: 본 프로그램은 문제 해결을 위한 주요 개념만을 구현하였으며 다양한 입력 데이터 형식으로 발생할 수 있는 오류에 대한 예외 처리는 구현되어 있지 않습니다.