대화형프로그래밍
과제3
도서 대출 반납 프로그램

학과: 수학과
학번: 202021224
이름: 주민찬
반: 대화형프로그래밍
작성일: 2020.06.27.

1. 요구사항 분석 결과

서비스 번호를 제시하고 사용자로부터 서비스 번호를 입력받아, 대출 또는 반납을 실행해주는 프로그램

2. 데이터 설계 결과

대출 권수는 3권 이하로 하고 날짜를 입력받을 때 월은 영어 대문자의 앞의 3글자만 입력 가능하고 년, 일은 오류가 없다고 가정, 대출한 도서명과 날짜는 딕셔너리에 저장. 반납은 대출현황을 먼저 출력한다. 대출일과 반납일의 일수를 윤년을 고려하여 계산. 대출일과 반납일의 일수가 50일이 넘을시에 연체일도 같이 출력.

3. 알고리즘

(1) 대출

-> 시작
-> 서비스 번호 1 입력
-> 대출일 입력
-> 대출일 오류시 오류메세지 출력 후 다시 입력
-> 오류가 아닐시 대출 권수입력
-> 대출권수가 3개 초과일시 오류메세지 출력 후 다시 입력
-> 오류가 아닐시 도서명 입력
-> 계속 진행여부
-> 종료

(2) 반납

-> 시작

-> 서비스 번호 2 입력

-> 현재 대출 현황 출력

-> 반납일 입력

-> 반납일 오류시 오류메세지 출력 후 다시 입력

-> 오류가 아닐시 반납할 도서명 입력

-> 도서의 대출일과 반납일 일수 계산 후 출력

-> 일수가 50일 초과일 경우 연체일 계산 후 출력

-> 반납 완료 메시지 출력

-> 계속 진행여부

-> 종료

(3) 서비스 번호 오류

-> 시작

-> 서비스 번호 1, 2가 아닐시 오류메세지 출력 후 다시진행

-> 서비스 번호 오류가 안 날때까지 두 번째 반복

-> 서비스 번호 오류 아닐시 (1) 또는 (2) 실행

*main()함수

```
while 1 :                                          # main()함수
    if cont == 'n' :
        break
    print('₩n[ 1. 대출, 2. 반납 ]')
    service = input('>>> 서비스 번호를 선택하세요 : ')
    if service == '1' :
        borrow()
        cont = input('>>> 계속 하시겠습니까?(y 또는 n) : ')
    elif service == '2' :
        if len(book_list) == 0 :
            print('₩n반납할 도서가 없습니다.')
            continue
        returns()
        cont = input('>>> 계속 하시겠습니까?(y 또는 n) : ')
    else :
        print('서비스 번호를 잘못 입력했습니다.')
```

*대출함수

```
def borrow() :                          # 대출 함수
    while 1 :
            print('')
            borrow_day = input('>>> [대출일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : ')
            borrow_day = borrow_day.split()
            if bool(borrow_day[1] in month) == True :
                borrow_day[1] = str(month[borrow_day[1]])
                borrow_day = borrow_day[0]+'년 '+borrow_day[1]+'월 '+borrow_day[2]+'일'
                while 1 :
                    count = int(input('>>> 대출 권수 입력 : '))
                    if count + len(book_list) > 3 :
                        print('대출 가능한 권수를 초과했습니다.')
                    else :
                        break
                for x in range(count) :
                    borrow_book = input('>>> 대출할 도서명 입력 : ')
                    book_list[borrow_book] = borrow_day
                print('')
                break
            else :
                print('월 입력이 잘못되었습니다.')
                continue
```

*반납함수

```
def returns() :                     # 반납 함수
    while 1 :
            print('₩n'+'*'*19 + '현재 대출 현황' + '*'*19)
            key = list(book_list.keys())
            value = list(book_list.values())
            for x in range(len(book_list)) :
                print('[{}] 대출일: {}, 도서명: {}' .format(x+1, value[x], key[x]))
            print('*'*52, '₩n')
            while 1 :
                return_day = input('>>> [반납일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : ')
                return_day = return_day.split()
                if bool(return_day[1] in month) == True :
                    return_day[1] = str(month[return_day[1]])
                    return_day = return_day[0]+'년 '+return_day[1]+'월 '+return_day[2]+'일'
                    return_book = input('>>> 반납할 도서명 입력 : ')
                    print('₩n< {} > 도서의 대출일은 {}이고, 반납일은 {}입니다.' .format(return_book, book_list[return_book], return_day))
                    day = count_day(int(book_list[return_book][:4]), int(book_list[return_book][6:8]), int(book_list[return_book][10:12]),
                                    int(return_day[:4]), int(return_day[6:8]), int(return_day[10:12]))
                    if day > 50 :
                        late_day = day-50
                        print('대출 기간은 총', str(day)+'일입니다.')
                        print('₩n'+str(late_day)+'일이 연체되었습니다.')
                        print('반납이 완료되었습니다.')
                    else :
                        print('대출 기간은 총', str(day)+'일입니다.')
                        print('₩n반납이 완료되었습니다.')
                    del book_list[return_book]
                    print('')
                    break
                else :
                    print('월 입력이 잘못되었습니다.')
                    continue
            break
```

*윤년 계산 함수

```
def leap_year(year) :                       # 윤년 계산 함수
    if year%4 == 0 :
        if year%100 == 0 and not(year%400 == 0) :
            return False
        else :
            return True
    else :
        return False
```

*일수 계산 함수

```
def count_day(Y1, M1, D1, Y2, M2, D2) :     # 일수 계산 함수
    if leap_year(Y1) == True and leap_year(Y2) == True :
        count = 1
        n = 0
        while Y1<Y2 :
            Y1 += 4
            if leap_year(Y1) == True :
                count += 1
            else :
                continue
            n += 1
        Y1 -= 4*n
        if M1 > 2 :
            day1 = (M1-1)*30 + M1//2 - 1 + D1
        else :
            day1 = (M1-1)*31 + D1
        if M2 > 2 :
            day2 = (M2-1)*30 + M2//2 - 1 + D2
        else :
            day2 = (M2-1)*31 + D2
        total_term = day2 + 365*(Y2-Y1)+count - day1-1
        return total_term
    elif leap_year(Y1) == True and leap_year(Y2) == False :
        count = 1
        n = 0
        while Y1<Y2 :
            Y1 += 4
            if leap_year(Y1) == True :
                count += 1
            else :
                continue
            n += 1
        Y1 -= 4*n
        if M1 > 2 :
            day1 = (M1-1)*30 + M1//2 - 1 + D1
        else :
            day1 = (M1-1)*31 + D1
        if M2 > 2 :
            day2 = (M2-1)*30 + M2//2 - 2 + D2
        else :
            day2 = (M2-1)*31 + D2
        total_term = day2 + 365*(Y2-Y1)+count - day1-1
        return total_term
    elif leap_year(Y1) == False and leap_year(Y2) == True :
        count = 0
        n = 0
        while Y1<Y2 :
            Y2 -= 4
            if leap_year(Y2) == True :
                count += 1
            else :
                continue
            n += 1
        Y2 += 4*n
        if M1 > 2 :
            day1 = (M1-1)*30 + M1//2 - 2 + D1
        else :
            day1 = (M1-1)*31 + D1
        if M2 > 2 :
            day2 = (M2-1)*30 + M2//2 - 1 + D2
        else :
            day2 = (M2-1)*31 + D2
        total_term = day2 + 365*(Y2-Y1)+count - day1-1
        return total_term
    else :
        count = 0
        n = 0
        while Y1<Y2 :
            Y1 += 1
            if leap_year(Y1) == True :
                count += 1
            else :
                continue
            n += 1
        Y1 -= n
        if M1 > 2 :
            day1 = (M1-1)*30 + M1//2 - 2 + D1
        else :
            day1 = (M1-1)*31 + D1
        if M2 > 2 :
            day2 = (M2-1)*30 + M2//2 - 2 + D2
        else :
            day2 = (M2-1)*31 + D2
        total_term = day2 + 365*(Y2-Y1)+count - day1-1
        return total_term
    print(total_term)
```

*대출일, 반납일 월 오류 계산

month 딕셔너리를 이용하여 입력한 월이 month안에 없으면 오류 발생

*대출권수 오류 계산

도서명 딕셔너리의 수를 len()함수를 이용하여 계산하고 대출할 대출권수를 더하여서 3을 초과 할시 오류 발생

*일수 계산 메커니즘

대출일과 반납일의 각 달과 일을 이용하여 같은 해 1월 1일부터의 일수를 계산 만약 대출일이 2020년 5월 16일일 경우 (5-1)*30+5//2+16-1으로 계산 만약 2019년처럼 윤년이 아닐 경우 1을 빼준다. 하지만 대출일이 1월 또는 2월인 경우 년도에 상관없이 '(달-1)*30+달//2 + 일'로 계산한다. 대출일과 반납일의 1월 1일 부터의 일수를 계산하고 나서 두 년도사이의 윤년의 개수를 세고 윤년이 아닌 경우 365 윤년일 경우 366을 더해서 계산한다.

*윤년 계산 메커니즘

윤년은 4의 배수인 년도를 말한다. 하지만 100으로 나누어 떨어지면서 400으로 나누어 떨어지지 않는 년도는 평년으로 한다.

*초기 변수들

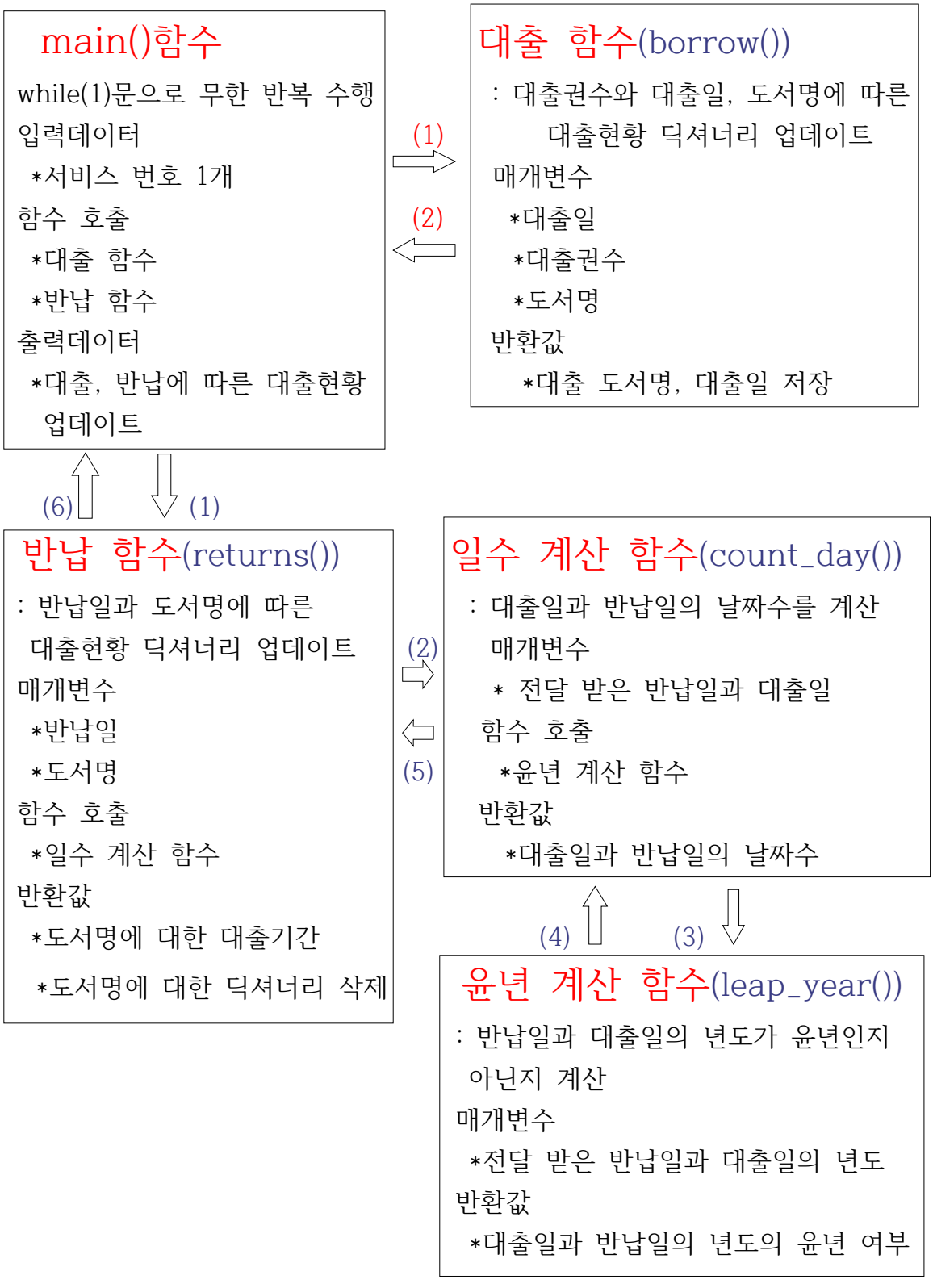
```
month = {'JAN':'01', 'FEB':'02', 'MAR':'03', 'APR':'04', 'MAY':'05', 'JUN':'06', 'JUL':'07', 'AUG':'08', 'SEP':'09', 'OCT':'10', 'NOV':'11', 'DEC':'12'}      # 달 입력 딕셔너리
book_list = {}                          # 책 입력 딕셔너리
cont = 'y'
```

(1) month = {'JAN':'01', 'FEB':'02', 'MAR':'03', 'APR':'04', 'MAY':'05', 'JUN':'06', 'JUL':'07', 'AUG':'08', 'SEP':'09', 'OCT':'10', 'NOV':'11', 'DEC':'12'}

(2) book_list = { }

(3) cont = 'y'

4. 함수 관계도


main()함수
while(1)문으로 무한 반복 수행
입력데이터
*서비스 번호 1개
함수 호출
*대출 함수
*반납 함수
출력데이터
*대출, 반납에 따른 대출현황 업데이트
(1)
(2)
대출 함수(borrow())
: 대출권수와 대출일, 도서명에 따른 대출현황 딕셔너리 업데이트
매개변수
*대출일
*대출권수
*도서명
반환값
*대출 도서명, 대출일 저장
(6)
(1)
반납 함수(returns())
: 반납일과 도서명에 따른 대출현황 딕셔너리 업데이트
매개변수
*반납일
*도서명
함수 호출
*일수 계산 함수
반환값
*도서명에 대한 대출기간
*도서명에 대한 딕셔너리 삭제
(2)
(5)
일수 계산 함수(count_day())
: 대출일과 반납일의 날짜수를 계산
매개변수
* 전달 받은 반납일과 대출일
함수 호출
*윤년 계산 함수
반환값
*대출일과 반납일의 날짜수
(4)
(3)
윤년 계산 함수(leap_year())
: 반납일과 대출일의 년도가 윤년인지 아닌지 계산
매개변수
*전달 받은 반납일과 대출일의 년도
반환값
*대출일과 반납일의 년도의 윤년 여부

5. 실행화면

```
[ 도서 대출 반납 프로그램 ]

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 1

>>> [대출일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2010 AUG 29
>>> 대출 권수 입력 : 1
>>> 대출할 도서명 입력 : Python

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 1

>>> [대출일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2016 FEB 04
>>> 대출 권수 입력 : 1
>>> 대출할 도서명 입력 : programming

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 1

>>> [대출일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2020 MAY 18
>>> 대출 권수 입력 : 2
대출 가능한 권수를 초과했습니다.
>>> 대출 권수 입력 : 1
>>> 대출할 도서명 입력 : 파이썬

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 2

**********현재 대출 현황**********
[1] 대출일: 2010년 08월 29일, 도서명: Python
[2] 대출일: 2016년 02월 04일, 도서명: programming
[3] 대출일: 2020년 05월 18일, 도서명: 파이썬
**********

>>> [반납일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2020 JUN 28
>>> 반납할 도서명 입력 : Python

< Python > 도서의 대출일은 2010년 08월 29일이고, 반납일은 2020년 06월 28일입니다.
대출 기간은 총 3591일입니다.

3541일이 연체되었습니다.
반납이 완료되었습니다.

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 2

**********현재 대출 현황**********
[1] 대출일: 2016년 02월 04일, 도서명: programming
[2] 대출일: 2020년 05월 18일, 도서명: 파이썬
**********

>>> [반납일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2020 JUN 28
>>> 반납할 도서명 입력 : 파이썬

< 파이썬 > 도서의 대출일은 2020년 05월 18일이고, 반납일은 2020년 06월 28일입니다.
대출 기간은 총 41일입니다.

반납이 완료되었습니다.

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 2

**********현재 대출 현황**********
[1] 대출일: 2016년 02월 04일, 도서명: programming
**********

>>> [반납일] 년, 월, 일을 입력하세요.(예:2020 JAN 05) : 2020 JUN 28
>>> 반납할 도서명 입력 : programming

< programming > 도서의 대출일은 2016년 02월 04일이고, 반납일은 2020년 06월 28일입니다.
대출 기간은 총 1606일입니다.

1556일이 연체되었습니다.
반납이 완료되었습니다.

>>> 계속 하시겠습니까?(y 또는 n) : y

[ 1. 대출, 2. 반납 ]
>>> 서비스 번호를 선택하세요 : 2

반납할 도서가 없습니다.
```

한권씩 대출하여 대출일이 다르게 대출

대출한 권수를 더해서 3을 초과할시 오류메세지 출력

반납일을 오늘로 하여 어플을 이용하여 계산 비교

연체일을 총 일수-50해서 계산

연체일이 없을시 그냥 출력

반납할 도서가 없을시 메시지 출력

6. 결과 평가

실행 했을 때 결과는 잘 나왔지만, 아쉬운 점은 코드가 너무 길어서 복잡했다. 코드를 더 간략히 하는 법을 알고 싶다.