

# LAB 03

2020-01 소프트웨어의 이해 01분반 / 조교 이경민

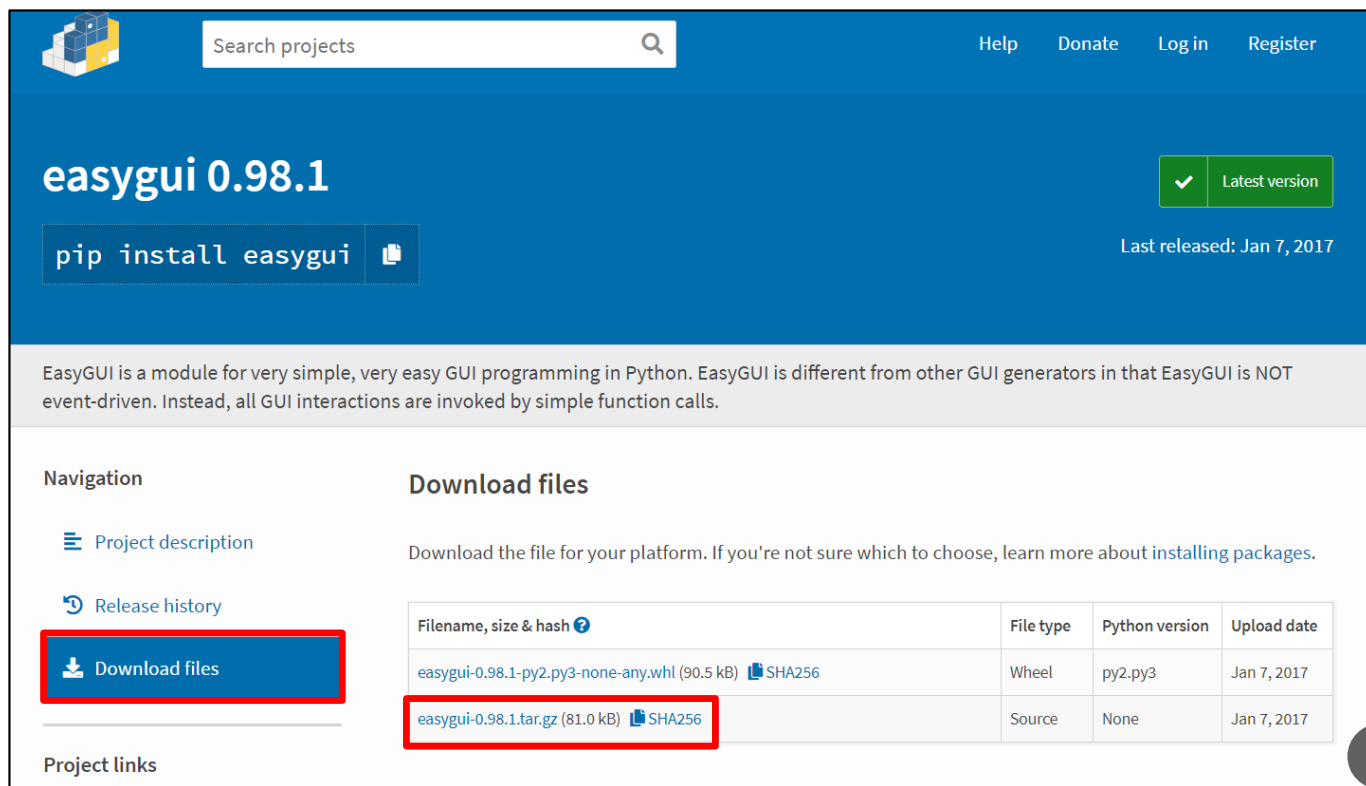
## LAB 03

---

- Easygui 설치
- 입력 / 출력
- 문자열
- GUI : easygui
- 파일처리
- 실습 과제

# 1. 설치 사이트에서 압축파일 다운로드

- 설치 사이트 : <https://pypi.org/project/easygui/>
- [Download files] 버튼 클릭하고 [easygui-0.98.1.tar.gz] 클릭



easygui 0.98.1

pip install easygui

Latest version

Last released: Jan 7, 2017



EasyGUI is a module for very simple, very easy GUI programming in Python. EasyGUI is different from other GUI generators in that EasyGUI is NOT event-driven. Instead, all GUI interactions are invoked by simple function calls.

Navigation

- Project description
- Release history
- Download files**

Download files

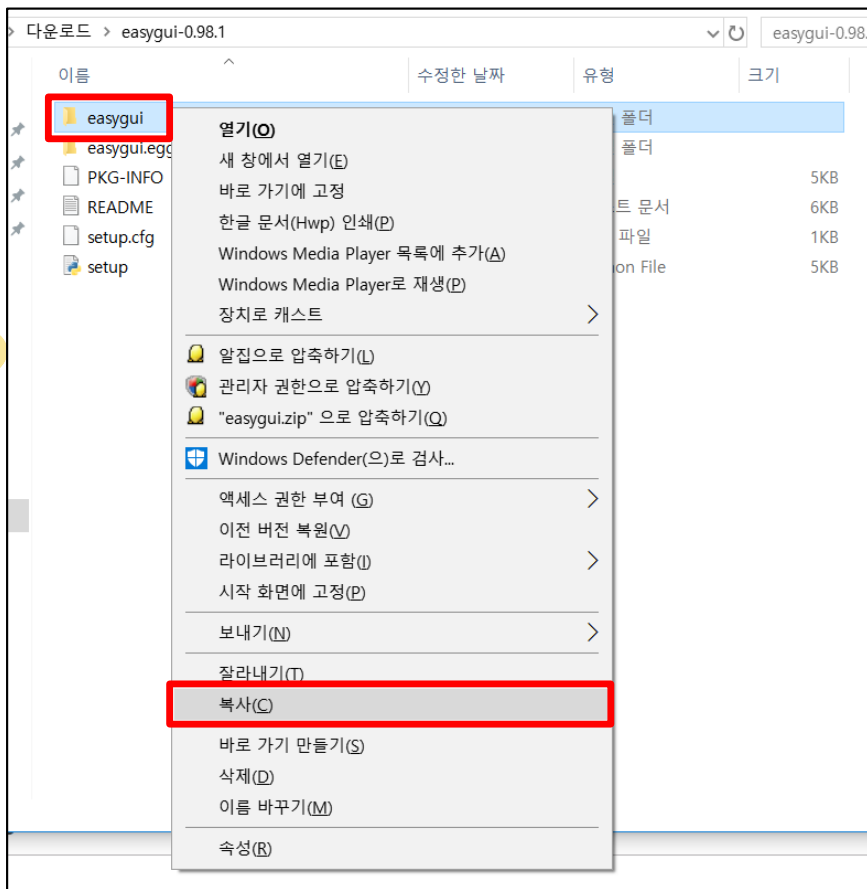
Download the file for your platform. If you're not sure which to choose, learn more about [installing packages](#).

Filename, size & hash	File type	Python version	Upload date
easygui-0.98.1-py2.py3-none-any.whl (90.5 kB) 	Wheel	py2.py3	Jan 7, 2017
<b>easygui-0.98.1.tar.gz (81.0 kB) </b>	Source	None	Jan 7, 2017

Project links

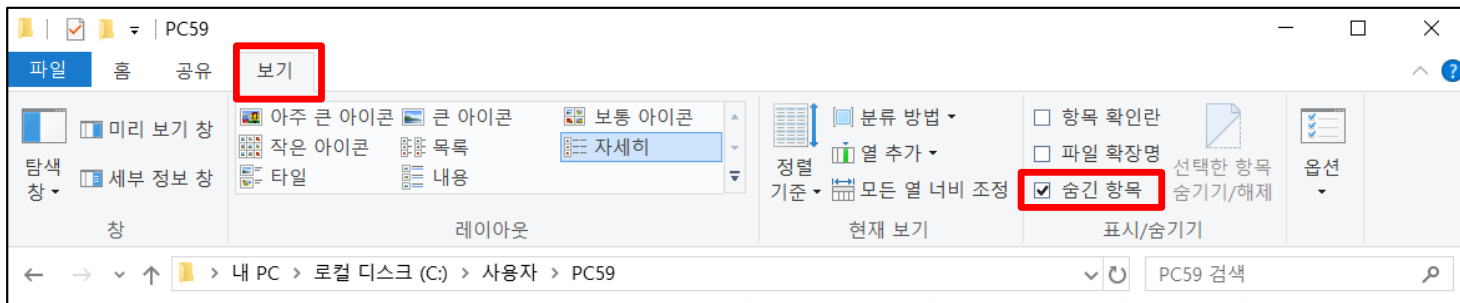
## 2. 다운받은 파일 압축 풀고 복사해두기

- [내 PC] - [다운로드] 폴더에서 easygui-0.98.1.tar 파일 찾아 압축풀기
- easygui-0.98.1 폴더 내에서 easygui 폴더 찾기
- easygui 폴더 통째로 복사해두기

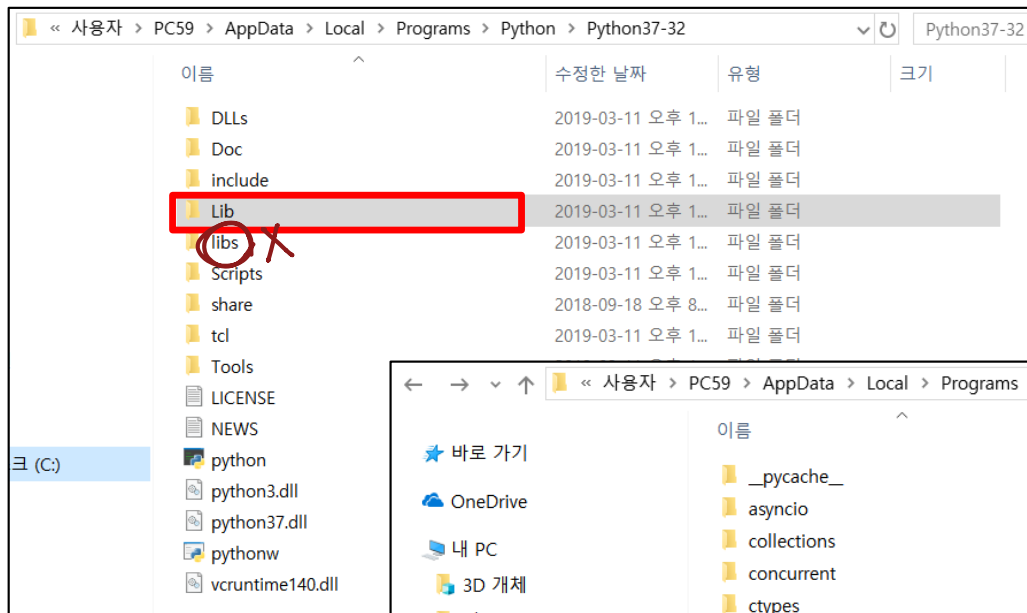


### 3. 파이썬 설치 경로의 Lib 폴더 밑에 붙여넣기

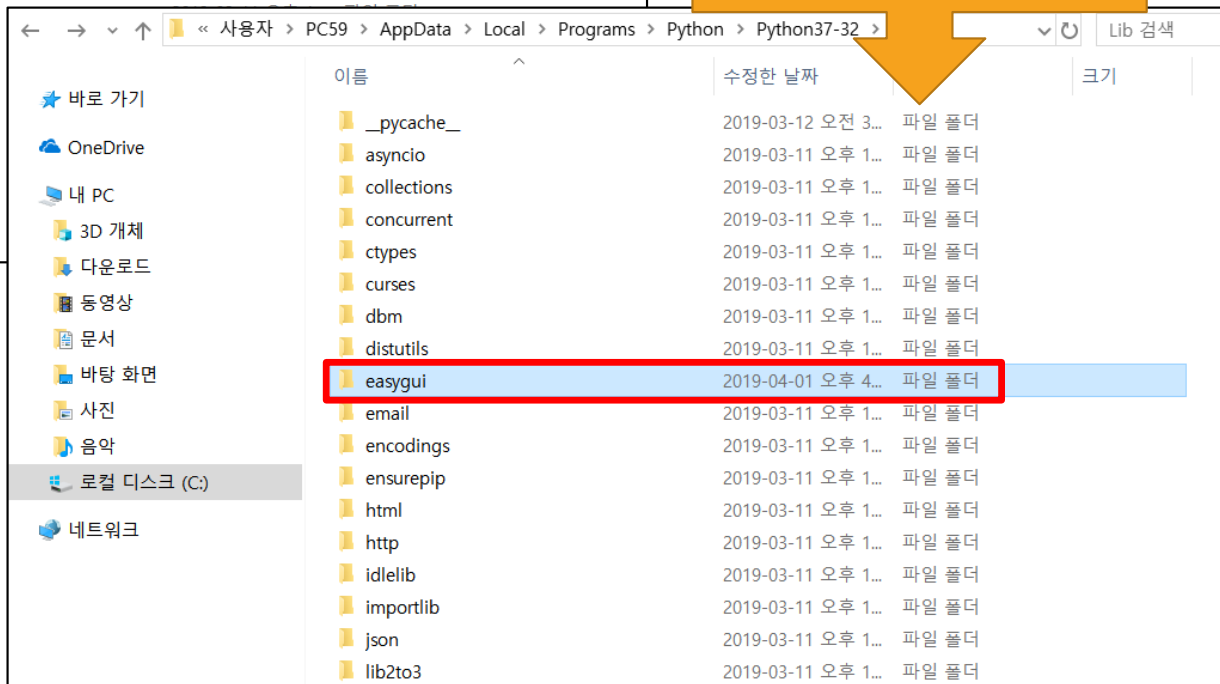
- 파이썬 설치 경로 (대부분)
  - C:\사용자(혹은 Users)\**본인사용자명**\AppData\Local\Programs\Python
- AppData 폴더가 안 보일 경우, 파일 탐색기에서 [보기] 탭 클릭 > 숨긴 항목에 체크



### 3. 파이썬 설치 경로의 Lib 폴더 밑에 붙여넣기



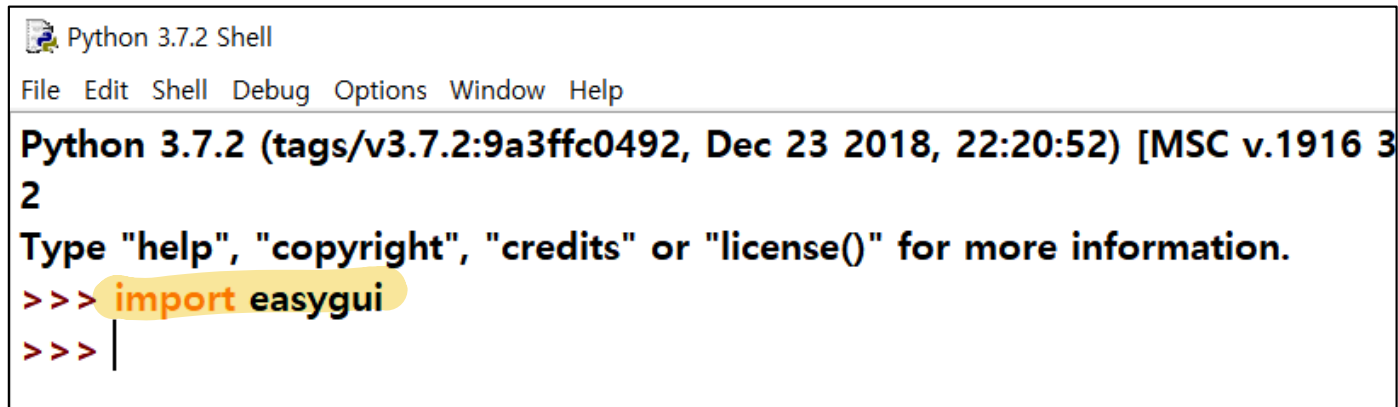
빈 공간 오른쪽 클릭하고  
붙여넣기 클릭



## 4. IDLE 켜다가 켜 뒤 확인

---

- import easygui 했을 때, 아무 메시지도 출력되지 않으면 성공!



The screenshot shows a Python 3.7.2 Shell window with a menu bar (File, Edit, Shell, Debug, Options, Window, Help). The main text area displays the Python version and build information: "Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32]". Below this, it prompts the user to type "help", "copyright", "credits" or "license()" for more information. The command prompt shows the user has entered ">>> import easygui" and the word "import" is highlighted in yellow. The next line shows ">>> |", indicating the command has been executed and the prompt is ready for the next input.

```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 32]
Type "help", "copyright", "credits" or "license()" for more information.
>>> import easygui
>>> |
```

# 입력 (input)

- 문자열과 정수, 실수를 구분하기!

input의 입력받는 경우 문자열이다

- 문자열 + 문자열 : 두 문자열을 붙여서 새로운 문자열 생성
- 문자열 \* 숫자 : 대상 문자열을 반복하여 새로운 문자열 생성
- cf) 정수 + 정수, 정수 \* 정수 : 우리가 흔히 알고 있는 사칙 연산 계산

```
>>> num = input()
5
>>> print(num)
5
>>> print(num + num)
55
>>> num = int(input())
5
>>> print(num + num)
10
>>> |
```

→ 문자열

→ 숫자

```
>>> num = input()
12
>>> print(num * 2)
1212
>>> num = float(input())
12
>>> print(num * 2)
24.0
>>> |
```

→ 문자열

→ 실수



# 출력 (output)

- print 문

- 여러 개의 값 출력 시 ,(comma) 사용
- **end** 옵션 - 두 출력 문장 사이에 들어갈 내용 지정
- 여러 개의 명령문을 한 줄에 작성할 경우 ; (세미콜론) 사용
- 두 개의 문자열 출력 : ' + '와 ' , '의 차이 비교

- **\n**

- '줄바꿈'을 의미
- print문의 출력 결과가 다음 줄에 나타남

```
>>> a = "hello"
>>> b = "world"
>>> print(a, end = '\'); print(b)
hello world✓
>>> print(a, end = '\n'); print(b)
hello✓
world
>>> print(a, b)
hello world
>>> print(a + b)
helloworld
```

↘ 줄바꿈

# \n 사용

---

- \n

- 특수 문자로 "줄바꿈"이라는 의미를 가짐, 마치 알파벳을 쓰듯 사용하면 됨.

*escape character*

```
>>> sentence = "Wish\nYou\nGood\nLuck"
>>> print(sentence)
Wish✓
You ✓
Good ✓
Luck
>>> |
```

## [참고] + 연산

- 동일한 타입의 피연산자에 대해서만 가능
  - 문자열 + 문자열 (o)
  - 숫자 + 숫자 (o)
  - 숫자 + 문자열 (x)
    - ⇒ 숫자를 문자열로 변환 or 문자열을 숫자로 변환 하여 사용
- , (콤마)로 문자열을 이어서 “출력” 하는 것은 출력 (print)문에서만 가능

```
>>> print("I am "+20)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    print("I am "+20)
TypeError: can only concatenate str (not "int") to str
>>> print("I am "+str(20))
I am 20
>>> print("I am ", "20", "years old")
I am 20 years old
>>> |
```

# 문자열 (string) format

- 받을 문자열 % 전달할 값
- 문자열에 변수 값 전달
  - 문자열 : %s , 정수 : %d
- 여러 개의 값 전달 : ( )안에 ','로 구분하여 전달

```
>>> s1 = "Python is easier than %s"
>>> s2 = "C"
>>> print(s1 % s2)
Python is easier than C
>>>
>>> s3 = "Python is %d times easier than %s"
>>> s4 = 10
>>> print(s3 % (s4, s2))
Python is 10 times easier than C
>>> |
```

54가 10.5인 경우 %d를 사용하면  
정수부분만 출력됨.

54가 10.5인 경우 %f를 사용하면  
정수부분만 출력됨.

```
>>> print("%d %s %f" % (10, 'abc', 5.0))
10 abc 5.000000
>>>
>>> a = 10
>>> b = 'abc'
>>> c = 5.0
>>> print("%d %s %f" % (a, b, c))
10 abc 5.000000
>>> |
```

# 문자열 자르기

- **split** 함수

- 문자열을 특정 내부 문자 기준으로 분할

default  
빈칸을기준으로

- ex) 특정 문장의 단어를 분리하고 싶을 때 : split(), split(" "), split(' ')

```
>>> sentence = "Have a nice day"
>>> sentence.split()
['Have', 'a', 'nice', 'day']
>>> sentence.split(" ")
['Have', 'a', 'nice', 'day']
>>> sentence.split(' ')
['Have', 'a', 'nice', 'day']
```

```
>>> fruits = "apple,banana,cherry,mango"
>>> fruits.split(",")
['apple', 'banana', 'cherry', 'mango']
>>> fruits = "apple/banana/cherry/mango"
>>> fruits.split("/")
['apple', 'banana', 'cherry', 'mango']
>>> |
```

# High-Low 게임

Random number (랜덤)

- 랜덤으로 생성된 숫자를 k번 안에 예측해서 맞추는 게임
- 사용자의 예측 값과 정답의 크기를 비교
  - "High" 또는 "Low" 로만 알려주고, k번의 기회가 끝나면 정답을 알려줌
- 3가지 파트로 구성
  - 1) 정답으로 랜덤 숫자를 생성
  - 2) 사용자로부터 예측 값을 입력 받음
  - 3) 정답과 예측 값을 비교하여 정답 여부와 크기 비교 결과를 알려줌

# High-Low 게임

- 1) 정답으로 랜덤 숫자를 생성

- 무작위 정수 발생 : random 모듈의 randint 함수 사용
  - 2개의 파라미터를 필요(최소값, 최대값)

```
import random
```

```
answer = random.randint(1, 100) # 1~100 사이의 난수 발생
```

```
num = 0
```

```
times = 6 # 6번으로 설정
```

반복을 실행하기 위해

1~100에 포함되지 않는 수를 처리

# High-Low 게임

- 2) 사용자로부터 예측 값을 입력 받음
  - 조건 1. 정답과 예측 값이 다르고 2. 시도 횟수가 6번보다 작을 때 까지

```
while num 0 != T answer T and times > 0:  
    num = int(input("값을 입력하세요: "))
```

- 3) 정답과 예측 값을 비교하여 정답 여부와 크기 비교 결과를 알려줌

num = answer  
이거나  
time = 0 일때  
while 문을  
break 하게 됨.

```
if num < answer:  
    print("정답보다 작습니다. ")  
elif num > answer:  
    print("정답보다 큼니다.")  
times = times - 1  
print(times, "번 기회가 남았습니다.")  
print()
```



# High-Low 게임

- 최종 결과 출력

```
if num == answer:
    print("정답입니다.")
else:
    print("더이상 기회가 없습니다. 정답은", answer)
```

맞았어요

Time=0안남아

- 실행 화면

값을 입력하세요: 50  
정답보다 작습니다.  
5 번 기회가 남았습니다.

값을 입력하세요: 75  
정답보다 작습니다.  
4 번 기회가 남았습니다.

값을 입력하세요: 87  
정답보다 작습니다.  
3 번 기회가 남았습니다.

값을 입력하세요: 94  
정답보다 큼니다.  
2 번 기회가 남았습니다.

값을 입력하세요: 89  
정답보다 작습니다.  
1 번 기회가 남았습니다.

값을 입력하세요: 91  
정답보다 작습니다.  
0 번 기회가 남았습니다.

더이상 기회가 없습니다. 정답은 92

# GUI (Graphic User Interface)

- 그래픽 사용자 인터페이스
  - 대화상자와 마우스를 사용하여 명령을 내리는 방식
- **easygui** 모듈 사용






```
Python 3.7.2 Shell
File Edit Shell Debug Options Window Help
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 22:20:52) [MSC v.1916 3
2
Type "help", "copyright", "credits" or "license()" for more information.
>>> import easygui
>>> |
```

## [참고] module 사용시 주의할 점

- 모듈을 import 할 때, **import 하는 모듈 이름과 같은 이름의 .py 파일이 해당 폴더 내에 존재하면 안된다.**
- 예) **import easygui** 를 사용하여 제대로 코딩했는데도 에러가 발생한다면, 혹시 해당 폴더 위치에 모듈 이름과 같은 **easygui.py** 파일이 있는지 확인한다.

```
import easygui
import turtle
```

가능합니다.

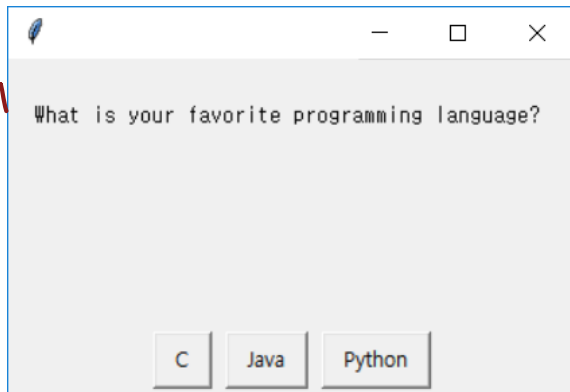
 easygui	✖	2019-04-10 오전 3...	Python File	1KB
 ex		2019-04-10 오전 3...	Python File	1KB
 turtle	✖	2019-04-10 오전 3...	Python File	1KB

# easygui 모듈

- 항상 **"import easygui"**
- **Buttonbox** : 메뉴 항목 버튼
- **Msgbox** : 출력 메시지

```
import easygui  
lang = easygui.buttonbox("What is your favorite programming language?",  
                           choices = ['C', 'Java', 'Python'])  
easygui.msgbox("You picked " + lang)
```

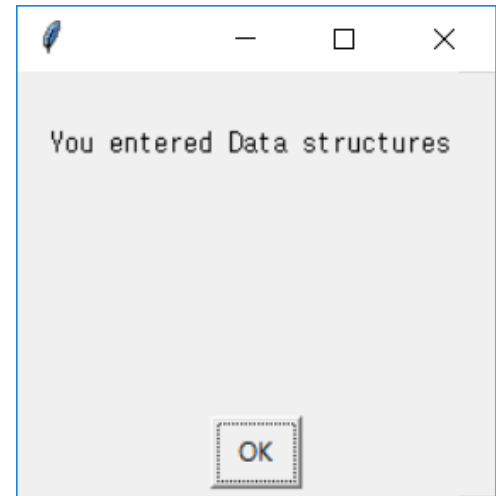
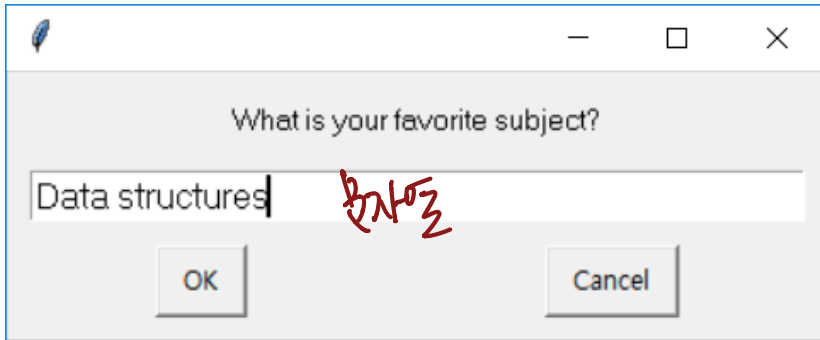
choice 리스트 만들기



# easygui 모듈

- **Enterbox** : 사용자로부터 문자열을 입력 받음

```
import easygui  
  
lang = easygui.enterbox("What is your favorite subject?")  
easygui.msgbox("You entered " + lang)
```



# easygui 모듈

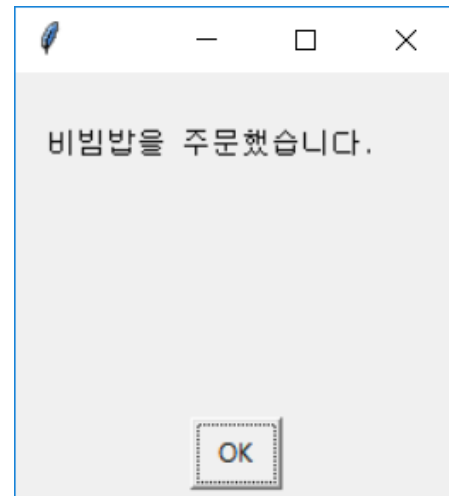
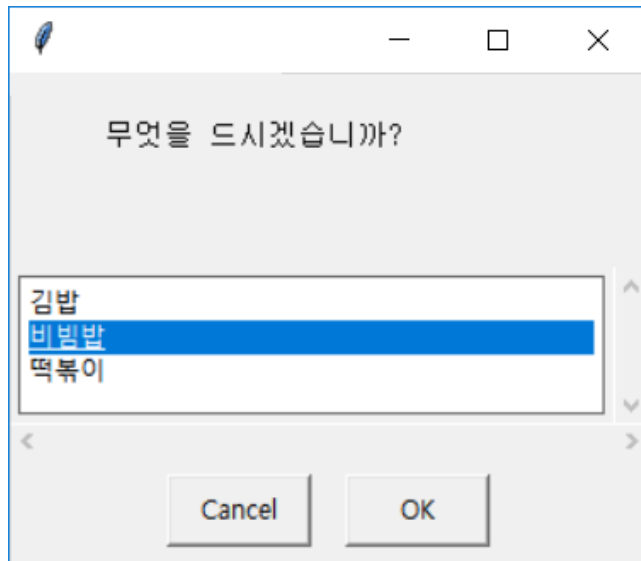
- **Choicebox** : 여러 개의 항목 리스트에서 하나의 항목을 선택

```
import easygui
```

미리 지정된 것 중 고름.

```
selection = ["김밥", "비빔밥", "떡볶이"]
```

```
reply = easygui.choicebox("무엇을 드시겠습니까?", choices = selection)  
easygui.msgbox(reply + "을 주문했습니다.")
```



# easygui 모듈 예제 : high-low 게임

- High-Low 게임을 easygui 모듈의 함수를 사용하여 대화 상자 형식으로 다시 작성

사용자에게 6번이면 안됨

- 1. 정답으로 랜덤 숫자를 생성
- 2. 사용자로부터 예측 값을 입력 받음
- 3. 정답과 예측 값을 비교하여, 정답 여부와 크기 비교 결과를 알려줌
- 4. 최종 결과 출력

```
import random
```

```
answer = random.randint(1, 100) # 1~100 사이의 난수 발생  
num = 0
```

```
times = 6 # 6번으로 설정
```

```
while num != answer and times > 0:  
    num = int(input("값을 입력하세요: "))  
    if num < answer:  
        print("정답보다 작습니다. ")  
    elif num > answer:  
        print("정답보다 큼니다.")  
    times = times - 1  
    print(times, "번 기회가 남았습니다.")  
    print()
```

```
if num == answer:  
    print("정답입니다.")
```

```
else:  
    print("더이상 기회가 없습니다. 정답은", answer)
```

# easygui 모듈 예제 : high-low 게임

---

- 1. 도전 횟수를 buttonbox로 입력 받음

```
import random
import easygui

answer = random.randint(1, 100)
num = 0
times = easygui.buttonbox("도전 횟수를 선택하세요", choices = ['5', '6', '7'])
times = int(times)
```

←  
문자열



## easygui 모듈 예제 : high-low 게임

- 2. 사용자의 **답을 enterbox를 통해서 입력 받음.**

enterbox > integerbox

```
while num != answer and times > 0 :  
    num = easygui.integerbox("1~100 사이의 숫자를 입력하세요. 도전 기회 = " + str(times))
```

숫자 입력을 integer로 바꿔서 줌.

문자열

## easygui 모듈 예제 : high-low 게임

---

- 3. 답이 정답보다 큰지, 작은지를 판단해서 결과를 msgbox로 출력

```
if num < answer:  
    easygui.msgbox(str(num) + "는 정답보다 작습니다.")  
elif num > answer:  
    easygui.msgbox(str(num) + "는 정답보다 큼니다.")  
times = times - 1
```

# easygui 모듈 예제 : high-low 게임

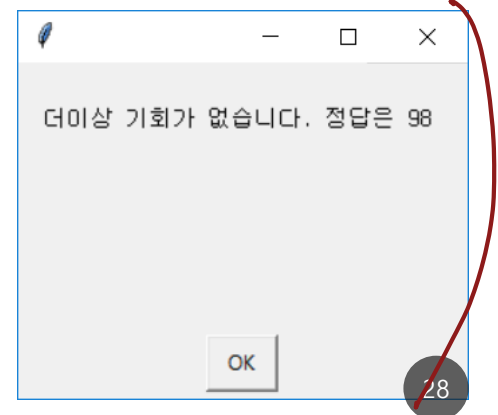
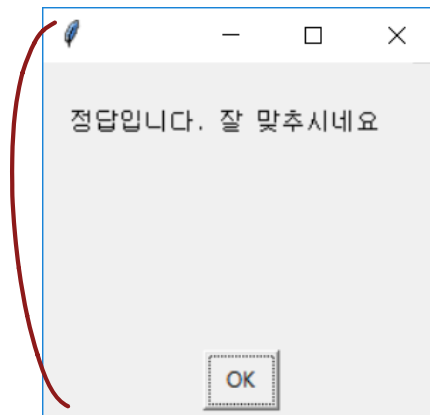
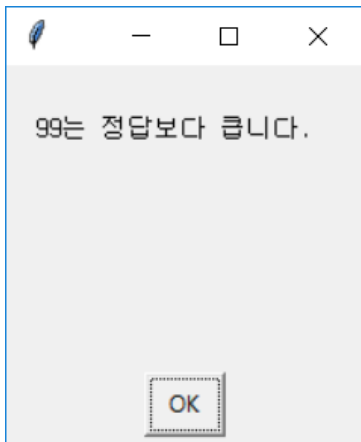
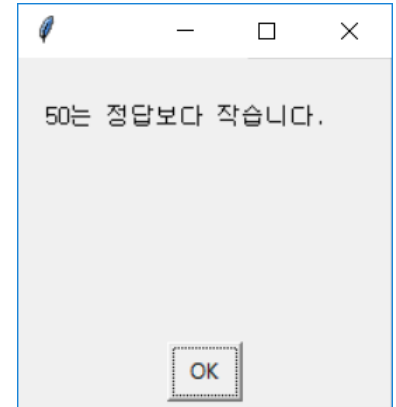
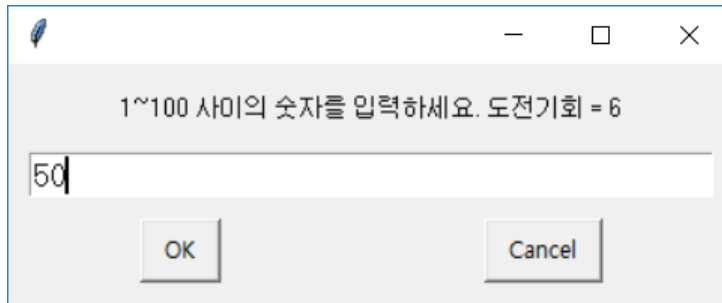
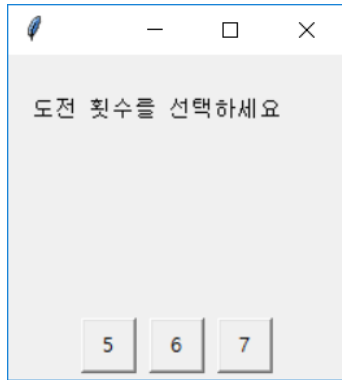
---

- 4. 최종결과 출력

```
if num == answer :  
    easygui.msgbox("정답입니다. 잘 맞추시네요")  
else :  
    easygui.msgbox("더이상 기회가 없습니다. 정답은 " + str(answer))
```

# easygui 모듈 예제 : high-low 게임

- 실행결과



## [참고] 왜 easygui box에서 , 로 문자열을 이루면 안되나요?

- 예) 함수  $g = f(x,y)$  *변수의 값으로 구분함.*
  - $f()$  안에  $x$ 와  $y$  총 2개의 값만 들어가야 하며 콤마(,) 로 각각  $x$ 와  $y$ 라는 것이 구분됨
- `easygui.buttonbox( 출력메세지 , choices = 버튼리스트 )`
  - 출력 메시지 부분에 문자열을 ,(콤마) 로 이루게 된다면,  
**해당 함수가 출력메세지와 choices 부분을 구분하지 못함!**

# 파일 처리

## • 1) 파일 읽기

- `open('파일이름')` : 파일을 읽기 모드로 열기
- `readline()` : 한 문장씩 읽어옴 **\n 까지.**
- `len('문자열')` : 문자열(string)의 길이를 알려줌
- `strip()` : 문자열에서 **양쪽 끝에 있는 공백과 \n 를 제거**해서 돌려줌  
**주의할 점!** 문자열 중간에 존재하는 공백과 \n를 제거하지 않음

```
>>> sentence = "Have\nA\nNice Day\n"
>>> print(sentence)
Have
A
Nice Day
>>> print(sentence.strip())
Have
A
Nice Day
```

# 파일 처리

## • 1) 파일 읽기

```
def read_file() :  
    f = open('memo.txt')  
    while True :  
        line = f.readline()  읽거나  
        if len(line) == 0 :  
            break  
        line = line.strip()  앞뒤공백자제  
        mymemo.append(line)  
    f.close()  닫거나  
  
mymemo = []  
read_file()  
for item in mymemo :  list  
    msg = item + '\n'  줄바꿈  
    print(msg)  
  
print(mymemo)
```

memo - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

orange  
apple  
candy  
noodle

memo.txt

===== RESTART: C

orange  
apple  
candy  
noodle

['orange', 'apple', 'candy', 'noodle']

# 파일 처리

## • 2) 파일 쓰기

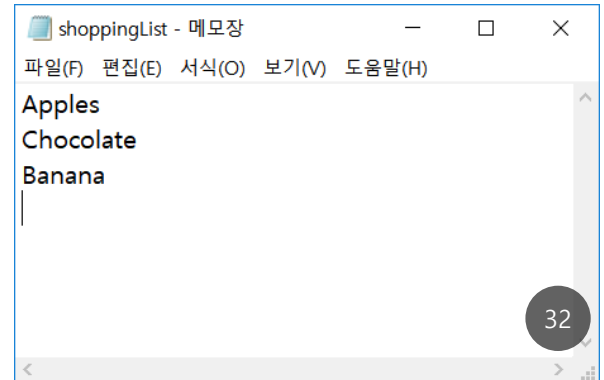
- `open('파일이름', 'w')`: 새로운 파일을 쓰기 모드로 열기
- `write()` : 문자열을 파일에 작성

```
def write_file() :  
    f = open('shoppingList.txt', 'w')  
    for item in mymemo :  
        msg = item + '\n'  
        f.write(msg)  
    f.close()  
  
mymemo = []  
while True :  
    print("Write an item to buy. Input a blank line to exit: ")  
    item = input()  
    if len(item) == 0 :  
        break  
    mymemo.append(item)  
print(mymemo)  
write_file()  
print(len(mymemo), "items are written into file")
```

파일 만들기 .

비교하다 .

```
===== RESTART: C:/SW_1911123/lab  
Write an item to buy. Input a blank line to exit:  
Apples  
Write an item to buy. Input a blank line to exit:  
Chocolate  
Write an item to buy. Input a blank line to exit:  
Banana  
Write an item to buy. Input a blank line to exit:  
  
['Apples', 'Chocolate', 'Banana']  
3 items are written into file
```





# 파일 처리

## • 2) 파일 쓰기

- `open('파일이름', 'a')`: 기존 파일을 쓰기 모드로 열기
- 뒤 내용에 추가 됨

덧붙이기. 'w'를 쓰면 기존내용을 덮어씀.

```
def write_file() :  
    f = open('shoppingList.txt', 'a')  
    for item in mymemo :  
        msg = item + '\n'  
        f.write(msg)  
    f.close()
```

```
mymemo = []  
while True :  
    print("Write an item to buy. Input a blank line to exit: ")  
    item = input()  
    if len(item) == 0 :  
        break  
    mymemo.append(item)  
print(mymemo)  
write_file()  
print(len(mymemo), "items are written into file")
```

===== RESTART: C:/SW\_1911123/lab

Write an item to buy. Input a blank line to exit:

Doll

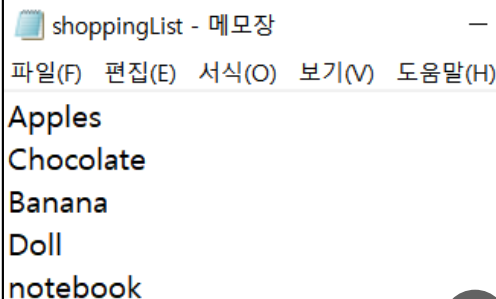
Write an item to buy. Input a blank line to exit:

notebook

Write an item to buy. Input a blank line to exit:

['Doll', 'notebook']

2 items are written into file



shoppingList - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

Apples  
Chocolate  
Banana  
Doll  
notebook

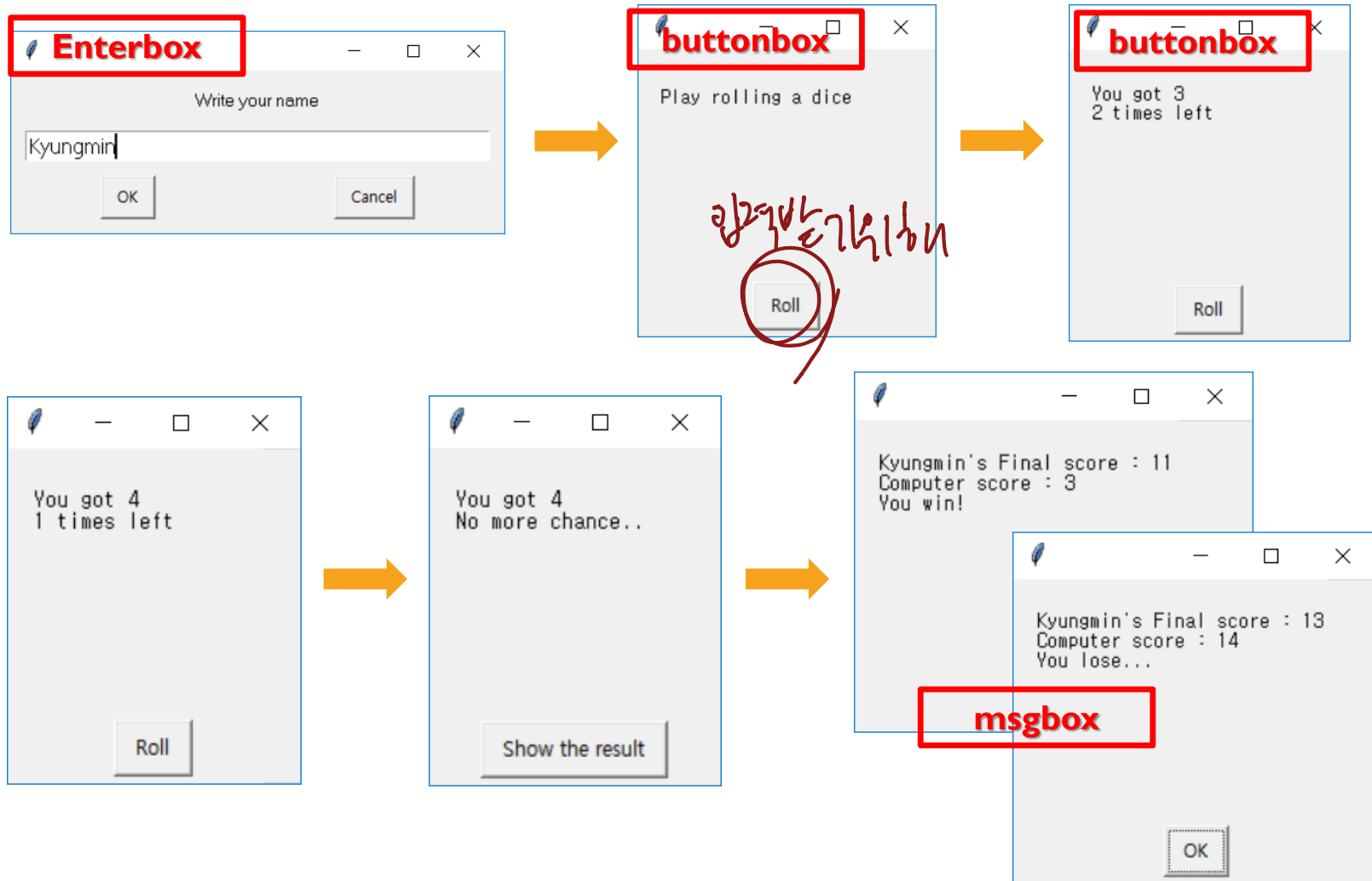
## LAB03 실습과제 1 (**dice.py**)

---

- 사용자로부터 이름을 입력 받고, 3번의 주사위를 던져서 얻은 총 합이 컴퓨터의 총 합보다 클 경우 이기는 게임
- 조건
  - 이름 입력 받기 : **Enterbox** 사용
  - 사용자가 주사위를 굴렀을 때 나오는 값 : **random** 모듈 사용 (1~6)
  - 첫 게임 시작 화면 & 점수 출력 화면은 모두 : **buttonbox** 사용
  - 컴퓨터 점수는 한 번에 구함 : **random** 모듈 사용 (3~18)
  - 마지막 결과 화면 출력 : **msgbox** 사용
- 다음 페이지의 실행 화면 참고

messagebox가 아니라.

# LAB03 실습과제 1 (**dice.py**)

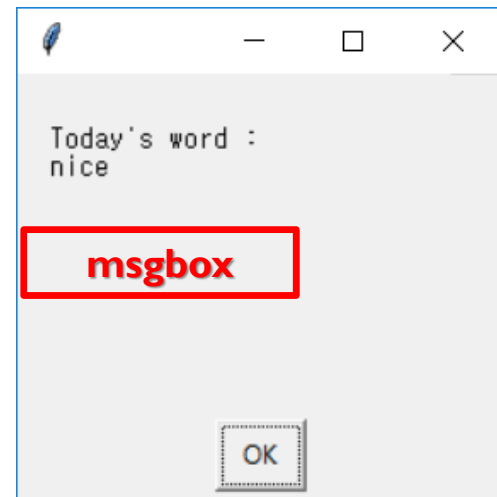
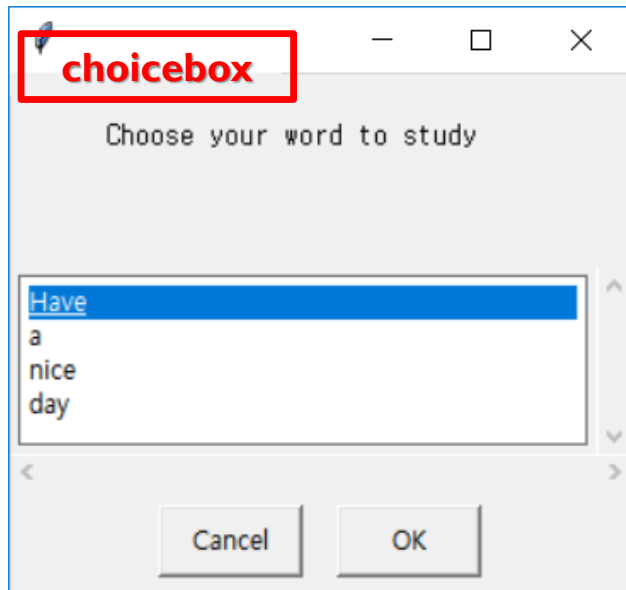
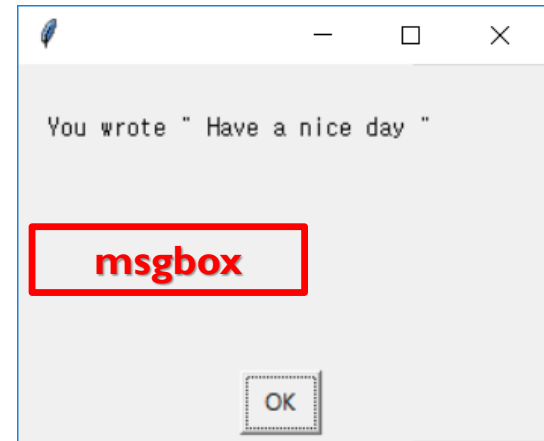
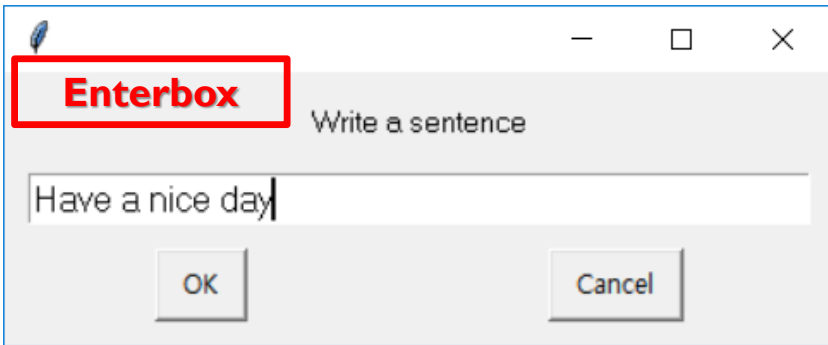


## LAB03 실습과제 2 (**words.py**)

---

- 문장을 입력하면, 단어로 분리해서 선택할 수 있는 프로그램을 작성
- 조건
  - 문장 입력 받기 : **enterbox** 사용
  - 단어 분리 : **split** 함수 사용
  - 단어 선택 : **choicebox** 사용
  - 결과들은 **msgbox**로 출력 (**단, msgbox에서는 %s를 꼭 사용한다**)
- 다음 페이지의 실행 화면 참고

## LAB03 실습과제 2 (**words.py**)



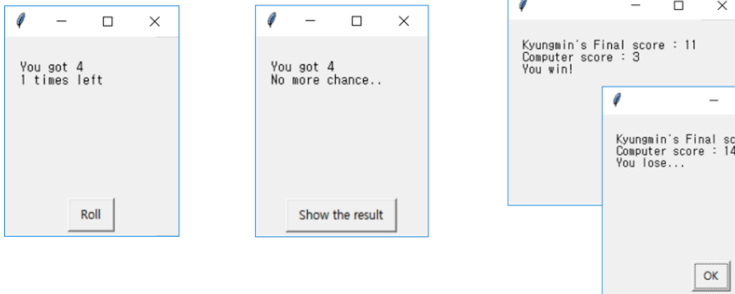
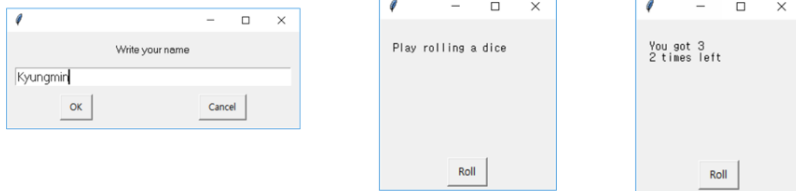
# 과제 보고서 예시

- 캡처 예시

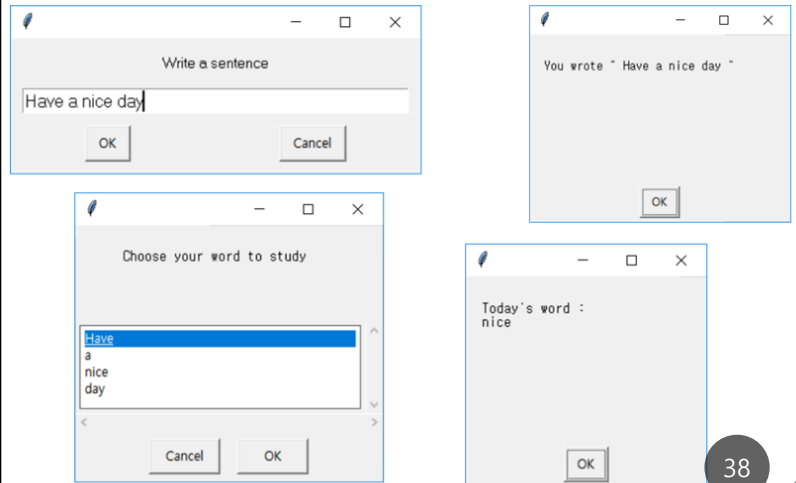
수업

-실행결과 화면 캡처

✓ 과제 1 (dice.py)



✓ 과제 2 (words.py)



# 과제 채점 기준·기한

---

- 과제 제출 기한

- 실습 다음주 화요일 오후 11시까지
- 4월 21일 화요일 오후 11시까지 제출

- 제출 장소

- 스노우보드 해당 주차 과제 제출 페이지에 업로드

- 추가 제출

- 제출기한 이후 24시간 이내 업로드 : 만점에서 20% 감점
- 그 이후는 받지 않음

- 표절 X

# 과제 제출 형식 & 질문 메일

---

- **제출물** : 소스파일(.py)과 과제보고서(.docx) 합친 **압축파일 제출**
  - **소스파일 이름** : 매 실습 과제 마다 ppt에 제시 예정
    - ex) test.py
  - **과제 보고서 양식** : 스노우보드에서 다운로드
- **(소스파일 + 과제보고서) 압축 파일 이름 : Lab03\_본인학번\_이름**
  - ex) Lab03\_1911123\_이경민.zip
- **이메일** : newkml22@gmail.com
- **질문 시 주의사항**
  - **과목, 분반, 이름, 학번** 알려주세요.
  - 몇 번 과제에서, 어떤 부분이 막혔는지, 어떤 과정인지 **설명**과 함께 보내주세요.
  - 답장까지 시간이 걸릴 수도 있으니 제출 과제 질문은 미리 해주세요!