from error import unexpectedError

from fileIO import fileIO, criticalQuestion

import random

from tkinter import \*

사용한 라이브러리는 위와 같다. tkinter를 통해 인터페이스를 만들고 random을 import했다.

코드의 디렉토리는 다음과 같다.

-main.py

-error.py

-fileIO.py

-image-사진들

def clear(window):

    for widget in window.winfo\_children():

        widget.destroy()

위 함수는 Tk에서 화면에 올라가있는 widget을 전부 삭제해주는 함수이다.

def asking(qList, character, ch, window):

    criterion = 8   # 점수 기준

    ok = False  # criterion 이상의 점수를 받은 캐릭터가 있는지 여부

    m = ''   # criterion 이상의 점수를 받은 캐릭터 이름

    # 질문 리스트를 돌며 질문하고 대답을 저장

    random.shuffle(qList)

위 코드는 asking 함수의 시작부분이다. 질문을 던지는 부분으로 기준점 8점 이상을 얻은 캐릭터가 있으면 질문을 중지한다. 그리고 이 8점은 질문 26개를 3으로 나누고 소수점을 버림한 숫자이다. 그리고 질문 리스트를 random.shuffle()을 통해 적절하게 섞어준다.

answer = IntVar()

    for i in range(len(qList)):

        Q = Label(window, text=qList[i][0], font=('궁서체',20))

        Q.pack()

        yes = Radiobutton(window, text='맞다.', variable=answer, value=1, command=lambda : clear(window))

        no = Radiobutton(window, text='아니다.', variable=answer, value=0, command=lambda : clear(window))

        dontknow = Radiobutton(window, text='모른다.', variable=answer, value=-1, command=lambda : clear(window))

        yes.pack()

        no.pack()

        dontknow.pack()

        yes.wait\_variable(answer)

        if answer.get() == 1 or answer.get() == 0:

            for j in range(0,len(character)):

                if answer.get() == qList[i][j + 2]:

                    character[j][1] += 1

                    if character[j][1] >= criterion:

                        ok = True

                        m = j

                else:

                    character[j][1] -= 1

                character[j][0][i] = answer.get()

        elif answer.get() == -1:

            continue

        else:

            raise unexpectedError('예기치 않은 오류 발생!')

위 코드가 본격적으로 질문을 하는 함수이다. Label을 통해 질문을 보여주고 밑에 Radiobutton을 통해 선택지를 받는다. 이 때 wait\_variable 메소드를 통해 버튼이 눌릴 때까지 기다린다. 그리고 밑에 대답을 해석하는 부분이다. 그리고 혹시 모를 에러 탐지를 위해 raise unexpectedError를 넣어주었다. 이 에러는 error.py에 있다.

        # 기준 점수 이상을 받은 캐릭터가 있고 동점이 없으면 출력, 동점이 있으면 계속 진행

        if ok:

            highestScore = []   # 기준 점수 이상을 받은 모든 캐릭터

            for j in range(len(character)):

                if character[m][1] == character[j][1]:

                    highestScore.append(j)  # 해당 캐릭터의 인덱스

위 코드는 일반적인 질문을 했을 때, 질문이 모두 끝나기 전, 기준 점수를 넘은 경우에 실행이 된다. 이 부분 탐지는 bool ok를 통해 확인한다. 그리고 highestScore list에 기준 점수 이상을 받은 모든 캐릭터의 인덱스를 넣는다.

            if len(highestScore) == 1:

                window.geometry('1000x800')

                photo = PhotoImage(file="./image/"+ch[highestScore[0]]+".png")

                targetPhoto = Label(image=photo)

                targetPhoto.image = photo

                targetPhoto.pack()

                target = Label(window,text=ch[highestScore[0]], font=('궁서체',20))

                target.pack()

                var = IntVar()

                finish = Button(window,text="끝내기", width=5,height=5, command=lambda: var.set(1))

                finish.pack()

                finish.wait\_variable(var)

위 코드는 만약 동점자가 없는 경우이다. 그럴 때에는 window 창 크기를 다시 늘려주고 해당 캐릭터 사진과 이름을 출력해주고 끝낸다.

else:

                # critical 질문 시작

                criticalQ = criticalQuestion()

                tmp = list(highestScore)

                answer = IntVar()

                for k in tmp:

                    Q = Label(window, text=criticalQ[k][0], font=('궁서체',20))

                    Q.pack()

                    yes = Radiobutton(window, text='맞다.', variable=answer, value=1, command=lambda : clear(window))

                    no = Radiobutton(window, text='아니다.', variable=answer, value=0, command=lambda : clear(window))

                    dontknow = Radiobutton(window, text='모른다.', variable=answer, value=-1, command=lambda : clear(window))

                    yes.pack()

                    no.pack()

                    dontknow.pack()

                    yes.wait\_variable(answer)

                    if answer.get() == 1:

                        window.geometry('1000x800')

                        photo = PhotoImage(file="./image/"+ch[k]+".png")

                        targetPhoto = Label(image=photo)

                        targetPhoto.image = photo

                        targetPhoto.pack()

                        target = Label(window,text=ch[k], font=('궁서체',20))

                        target.pack()

                        var = IntVar()

                        finish = Button(window,text="끝내기", width=5,height=5, command=lambda: var.set(1))

                        finish.pack()

                        finish.wait\_variable(var)

                        return True

                    elif answer.get() == 0:

                        highestScore.remove(k)

                    else:

                        break

                if len(highestScore) == 0:

                    return asking(qList,character,ch,window)   # 만약 끝까지 정답 안 나오면 재시작

위 코드는 일정 기준을 넘은 점수를 받은 캐릭터가 여럿일 때 실행이 된다. 동점 캐릭터들을 구분하는 결정적인 질문들을 하고 그 답변을 통해 정답을 추리한다. 만약 모두 아니라면 recursion을 통해 asking 함수를 재실행하여 결과를 얻는다.

def akinator():

    # 위젯 전체 삭제 및 창사이즈 조절

    window.geometry('400x400')

    for widget in window.winfo\_children():

        widget.pack\_forget()

    qList = []  #[[질문,특징번호,질문개수,선택지1,선택지2,선택지3]]

    character = []  #[[특징번호가 index이고 수치가 value], 질문 총합 점수]

    # csv파일 접근하여 데이터 in

    fileIO(qList, character)

    ch = qList[0]

    ch = ch[2:]

    del qList[0]

    for i in range(len(qList)):

        for j in range(len(qList[i])):

            if qList[i][j].isdigit():

                qList[i][j] = int(qList[i][j])

    if(asking(qList, character, ch, window)):

        print('정상 종료')

        exit()

    else:

        for i in range(len(character)):

            character[i].append(i)

        sortedCharacter = sorted(character, key=lambda k : k[1], reverse=True)

        window.geometry('1000x800')

        photo = PhotoImage(file="./image/"+ch[sortedCharacter[0][2]]+".png")

        targetPhoto = Label(image=photo)

        targetPhoto.image = photo

        targetPhoto.pack()

        target = Label(window,text=ch[sortedCharacter[0][2]], font=('궁서체',20))

        target.pack()

        var = IntVar()

        finish = Button(window,text="끝내기", width=5,height=5, command=lambda: var.set(1))

        finish.pack()

        finish.wait\_variable(var)

main 함수에서 불리는 최초 실행 함수이다. 메인창에서 실행을 누르면 실행되는 함수로 각 변수와 데이터를 초기화하는 부분이다.