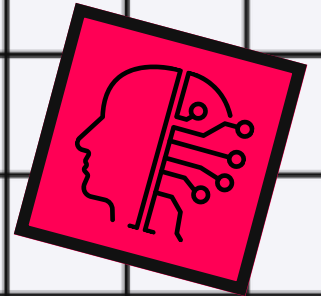


AI, 제대로 써보자!



미래를 위한 AI 활용 교육



▶ 안녕하세요!

HELLO!



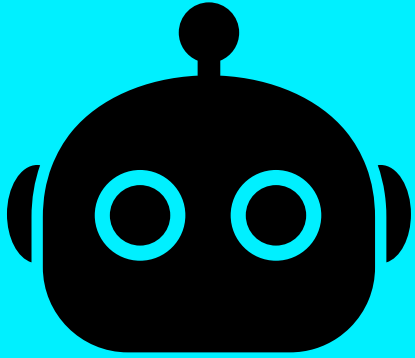
다양한 유·초·중·고등학교에서
소프트웨어 교육 봉사 진행

저희는 경기대학교
SWAT 봉사단입니다!

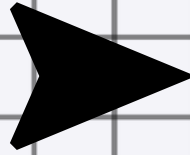


▶ 오늘 뭐 배우나요?

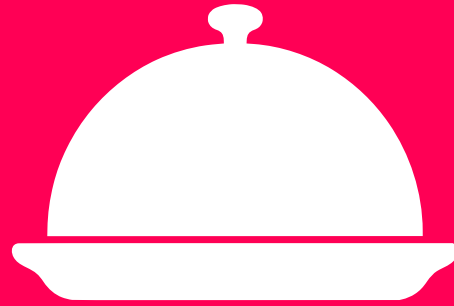
STAGE 1



AI 활용 교육

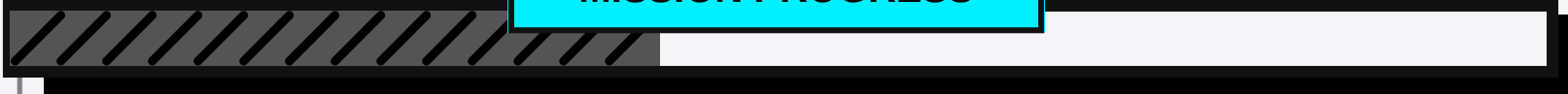


STAGE 2



서빙로봇 만들기

MISSION PROGRESS



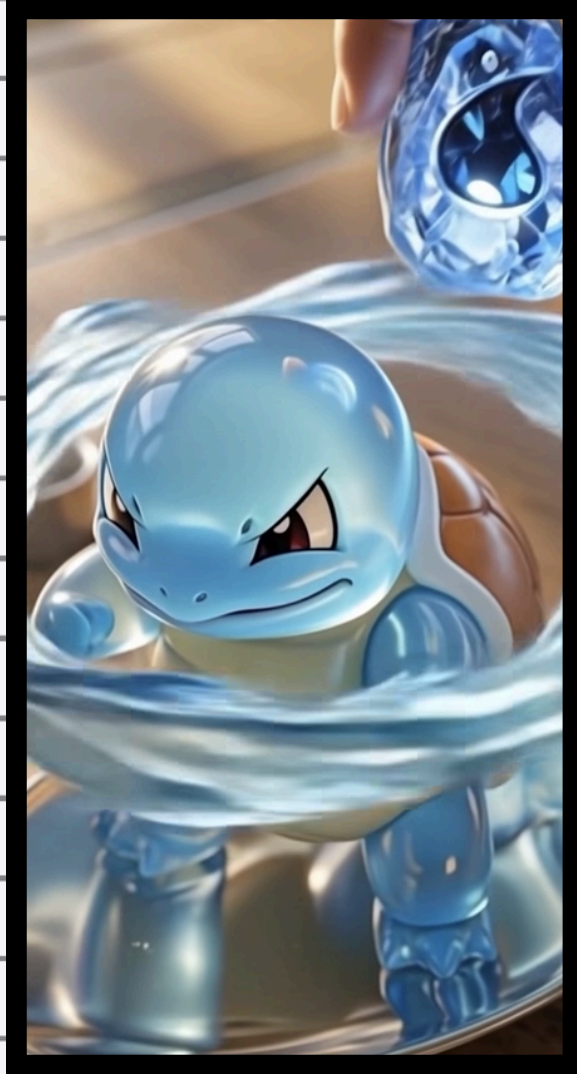
▶ 잠깐, 질문 하나!



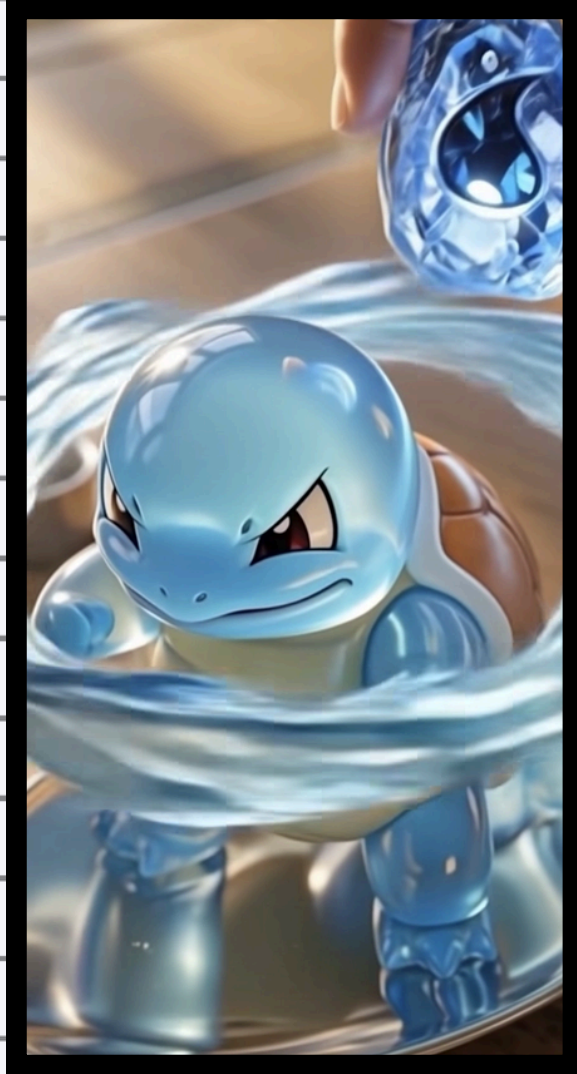
**“유튜브 숏츠나 인스타 릴스,
많이 보나요?”**



▶ 이거...진짜일까요?



▶ 이거...진짜일까요?

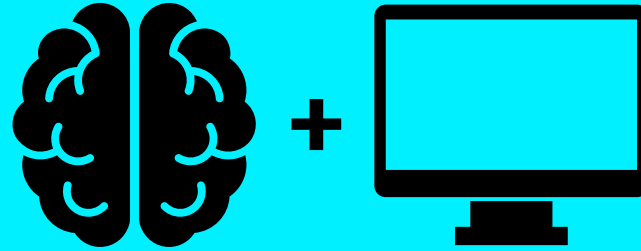


AI가 제작!



▶ AI가 뭔가요?

DEFINITION

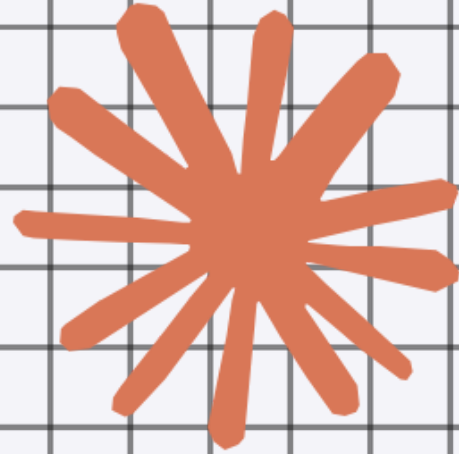
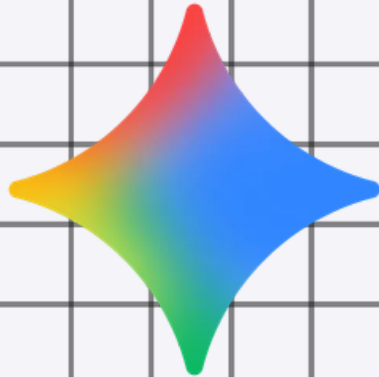


**ARTIFICIAL
INTELLIGENCE**

“사람처럼 생각하는 컴퓨터”

▶ AI를 써보셨나요?!

🔍 우리 주변 AI 찾아보기



▶ AI 관련 영상



▶ 이젠 미래가 아닙니다



제 1차 산업혁명



제 2차 산업혁명



제 3차 산업혁명



제 4차 산업혁명

“이젠 미래가 아니라, **현재**입니다“

▶ AI가 세상을 바꾸고 있어요

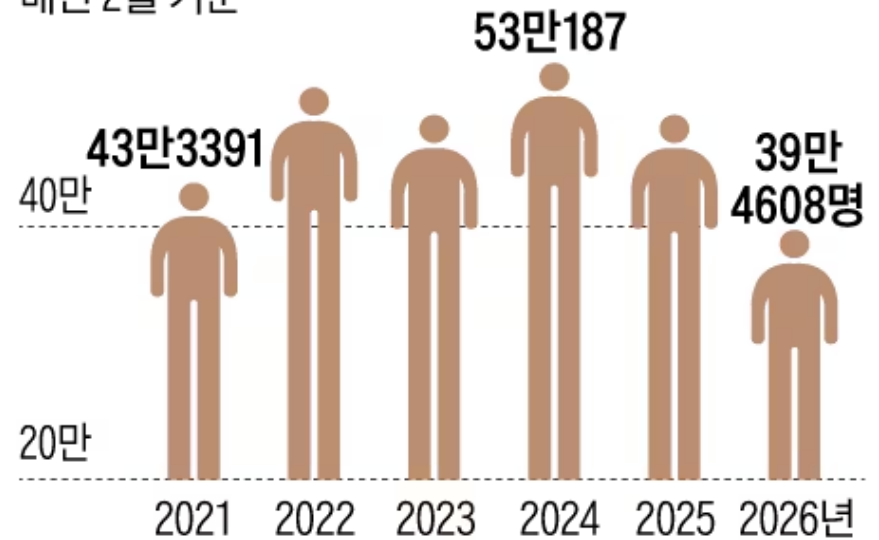
**AI가 실제로
일자리에 영향을
주고 있습니다**



자동화 및 대체되는 직업 증가

IMPACT

청년층 전문·과학·정보통신업 취업자 추이
매년 2월 기준



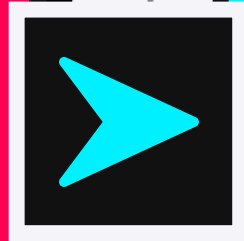
자료=국가데이터처

▶ 그래서 어떻게 해야 할까요?

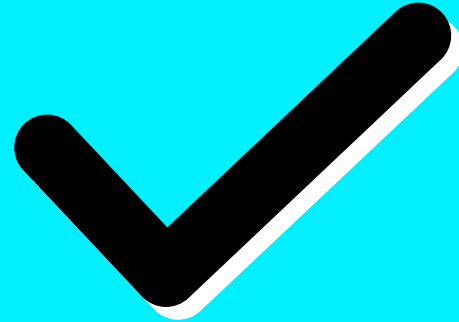
WRONG



“AI보다
똑똑해져야지”



CORRECT



“AI를
잘 활용해야지”

▶ 앞으로 가장 중요한 능력은?



AI 활용 능력

~~코딩보다, 영어보다, 수학보다~~

AI를 잘 쓰는 능력이 핵심입니다!

▶ AI를 잘 쓰는 방법: 질문의 힘! ⚡

질문 하나로 답변이 180도 달라집니다!

애매한 질문

?

답변도 애매하게... 🌤️



구체적인 질문



답변 퀄리티 폭발! 🚀

▶ 프롬프트와 프롬프팅이란?

프롬프트

AI에게 입력하는 질문이나 요청

+

프롬프팅

AI에게 원하는 답을 얻기 위해
프롬프트를 잘 작성하는 방법

"AI에게 어떻게 말해야 내가 원하는 답을 잘 받을까?"
를 고민하는 방법입니다!

▶ 꿀팁 1: AI와 상황극 하기! 🤖🎭

"너는 **OO**이야"

AI에게 역할을 먼저 정해주세요!



내 상황에 딱 맞는
답변을 받을 수 있어요!



"너는 **중학교 수학 선생님**이야.
기말고사 예상 문제를 내줘."



"너는 **대학생 강사**야.
중학생을 위해 쉽게 설명해줘."



"너는 **나의 단짝 친구**야.
고민을 들어주고 위로해줘."

▶ 꿀팁 2: 구체적으로 알려주기! 🎯

나쁜 예시

✘

“역사 요약해줘.”



좋은 예시

✔

“나는 중학생이고 역사 기말고사를 준비하고 있어.
중요한 사건은 시대 순서대로 정리해주고,
핵심 개념은 5개로 뽑아서 쉽게 설명해줘.
마지막에는 3줄 요약도 붙여줘!”

무엇을 해줘 + **어떻게** 해줘

▶ 꿀팁 3: 순서대로 요청하기! 📖

한 번에 다 시키면 AI도 당황해요! 😞

1 "너는 중학생 역사 선생님이야" (역할 부여)

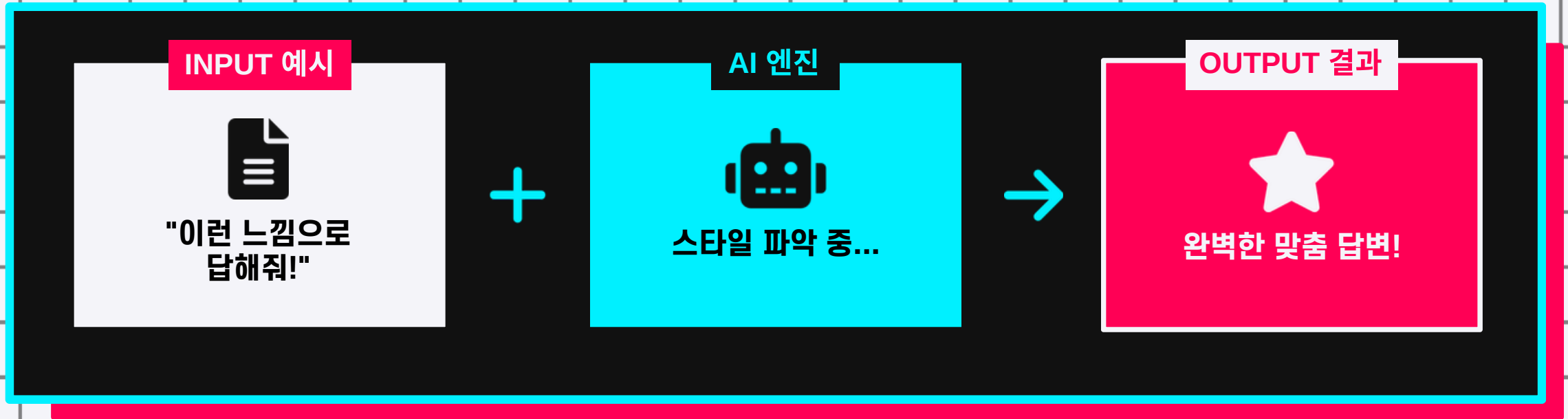
2 "이 파일에서 중요한 사건 5개만 뽑아줘"

3 "좋아! 각 사건의 원인과 결과를 표로 만들어줘"

4 "이 내용으로 예상 문제 5개만 만들어줘"

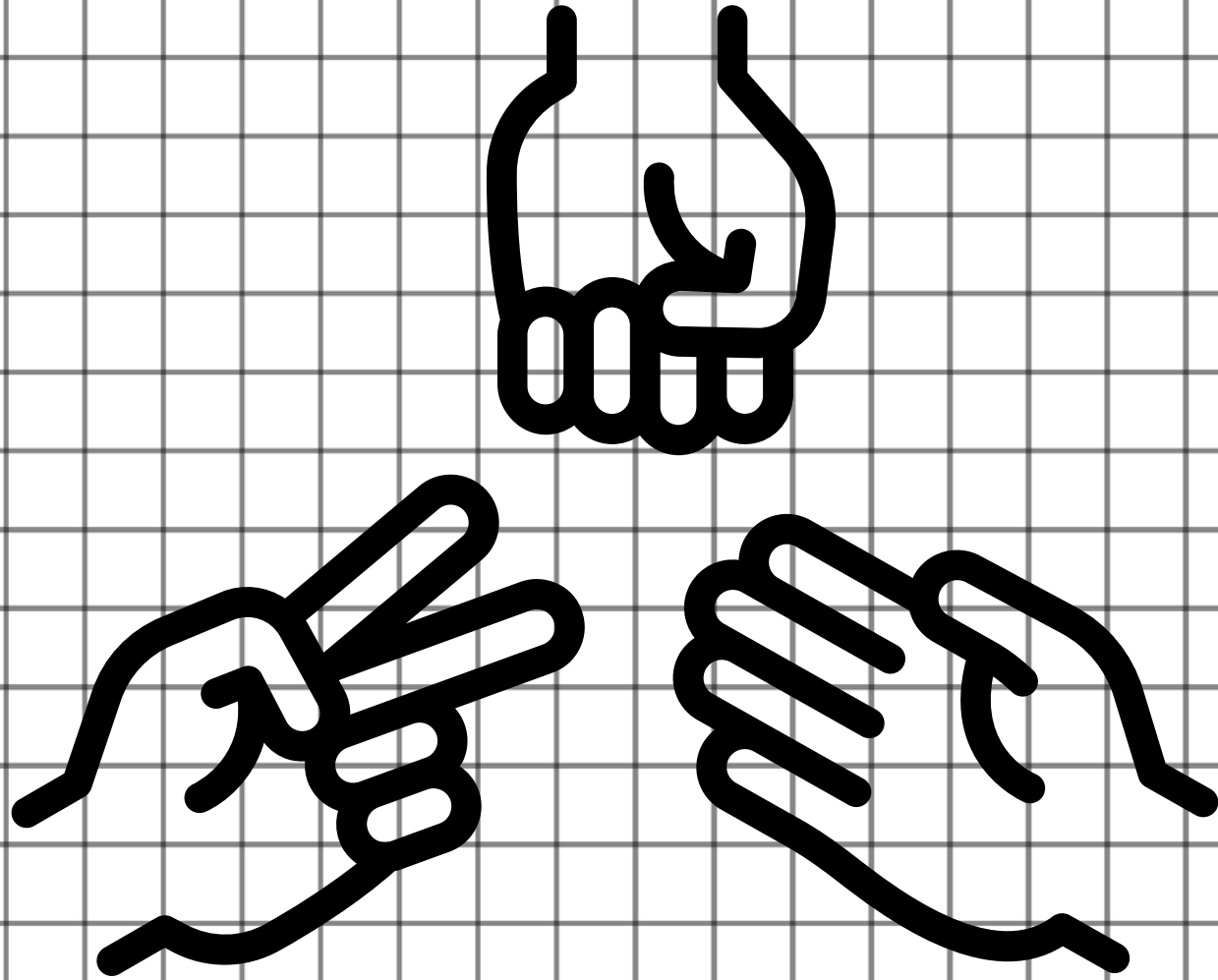
하나씩 차근차근! AI와 대화하세요 💬

▶ 꿀팁 4: 예시를 보여주기! 📝



"백 마디 설명보다, 한 번의 예시가 더 강력합니다!"

▶ 활용해보자!



▶ 활용해보자! – BAD

BAD PROMPT

가위바위보 게임 만들어줘

- Thought for 1s >
 - 가위바위보 게임을 HTML 파일로 만들겠습니다.
 - **Write** c:#SWAT#rps_game.html
357 lines

```
<!DOCTYPE html>
<html lang="ko">
<head>
```
 - [rps_game.html](#) 파일을 생성했습니다.
기능:
 - 가위 / 바위 / 보 버튼 클릭으로 플레이
 - 나 vs 컴퓨터 점수판 (승/무/패)
 - 선택 시 흔들리는 애니메이션
 - 최근 20판 전적 히스토리
 - 점수 초기화 버튼
 - 다크 그라디언트 UI
- 브라우저에서 파일을 열어 바로 플레이할 수 있습니다.

▶ 활용해보자! – GOOD

GOOD PROMPT

너는 중학생에게 Python을 가르치는 친절한 코딩 선생님이야.

나는 Python으로 간단한 가위바위보 게임 프로그램을 만들고 싶어.
아직 코딩을 많이 해보지 않은 중학생도 이해할 수 있게 만들어줘.

조건은 다음과 같아.

1. 사용자가 가위, 바위, 보 중 하나를 입력하게 해줘.
2. 컴퓨터는 랜덤으로 가위, 바위, 보 중 하나를 선택하게 해줘.
3. 사용자와 컴퓨터의 선택을 비교해서 승리, 패배, 무승부를 출력해줘.
4. 코드는 너무 어렵지 않게 작성해줘.
5. 각 코드 줄마다 무슨 의미인지 쉽게 주석을 달아줘.
6. 먼저 전체 코드부터 보여주고, 그다음 코드 작동 방식을 단계별로 설명해줘.

예시는 이런 느낌이면 좋겠어.

사용자 입력: 가위

컴퓨터 선택: 보

결과: 사용자가 이겼습니다!

▶ 활용해보자! – GOOD

전체 코드 + 단계별 설명 + 실행 예시
까지 ai가 제공!

가위바위보 게임 만들기 🖐️ 🖐️ 🖐️

전체 코드

```
import random # 컴퓨터가 랜덤으로 선택하려면 random 모듈이 필요해요

# 가위, 바위, 보 목록을 리스트로 만들어요
choices = ["가위", "바위", "보"]

# 사용자에게 입력을 받아요
user = input("가위, 바위, 보 중 하나를 입력하세요: ")

# 컴퓨터가 랜덤으로 하나를 선택해요
computer = random.choice(choices)

# 사용자와 컴퓨터의 선택을 화면에 보여줘요
print(f"사용자 입력: {user}")
print(f"컴퓨터 선택: {computer}")

# 무승부 확인: 둘의 선택이 같으면 무승부예요
if user == computer:
    print("결과: 무승부입니다!")

# 사용자가 이기는 경우 3가지를 확인해요
elif (user == "가위" and computer == "보") or \
      (user == "바위" and computer == "가위") or \
      (user == "보" and computer == "바위"):
    print("결과: 사용자가 이겼습니다!")

# 위 경우가 아니면 무조건 컴퓨터가 이긴 거예요
else:
    print("결과: 컴퓨터가 이겼습니다!")
```

단계별 설명

1단계 — 준비하기 (import random)

```
import random

random은 파이썬에 내장된 도구 상자예요.
이걸 불러와서 컴퓨터가 랜덤으로 선택할 수 있어요.
```

2단계 — 선택지 목록 만들기

```
choices = ["가위", "바위", "보"]

세 가지 선택지를 리스트라는 바구니에 담아주세요.
나중에 컴퓨터가 이 바구니에서 하나를 골라요.
```

3단계 — 사용자 입력 받기

```
user = input("가위, 바위, 보 중 하나를 입력하세요: ")

input()은 키보드로 글자를 입력받는 함수예요.
입력한 내용이 user 변수에 저장돼요.
```

4단계 — 컴퓨터가 랜덤 선택하기

```
computer = random.choice(choices)

random.choice()는 리스트에서 무작위로 하나를 골라줘요.
마치 컴퓨터가 물리 주사위를 굴리는 것과 같아요!
```

5단계 — 결과 비교하기

사용자 승리 조건 3가지

가위 vs 보	→	사용자 승
바위 vs 가위	→	사용자 승
보 vs 바위	→	사용자 승

if / elif / else 로 세 가지 경우를 순서대로 확인해요.

조건	결과
user == computer	무승부
위 승리 조건 중 하나 사용자 승	사용자 승
나머지 전부	컴퓨터 승

실행 예시

가위, 바위, 보 중 하나를 입력하세요: 가위
사용자 입력: 가위
컴퓨터 선택: 보
결과: 사용자가 이겼습니다!

▶ 활용해보자! – GOOD

점진적 업그레이드!

사용자가 가위, 바위, 보 말고 다른 값을 입력하면 다시 입력하게 코드를 수정해줘.

수정된 코드

```
import random # 컴퓨터가 랜덤으로 선택하려면 random 모듈이 필요해요

# 가위, 바위, 보 목록을 리스트로 만들어요
choices = ["가위", "바위", "보"]

# 올바른 값을 입력할 때까지 계속 반복해요
while True:
    user = input("가위, 바위, 보 중 하나를 입력하세요: ") # 사용자 입력을 받아요

    if user in choices: # 입력한 값이 choices 목록 안에 있으면
        break # 반복을 멈추고 아래로 내려가요
    else:
        print(f"{user}는 올바른 입력이 아니에요. 가위, 바위, 보 중 하나만 입력하세요!") # 경고 메시지를 보여줘요

# 컴퓨터가 랜덤으로 하나를 선택해요
computer = random.choice(choices)

# 사용자와 컴퓨터의 선택을 화면에 보여줘요
print(f"사용자 입력: {user}")
print(f"컴퓨터 선택: {computer}")

# 무승부 확인: 둘의 선택이 같으면 무승부예요
if user == computer:
    print("결과: 무승부입니다.")

# 사용자가 이기는 경우 3가지를 확인해요
elif (user == "가위" and computer == "보") or \
     (user == "바위" and computer == "가위") or \
     (user == "보" and computer == "바위"):
    print("결과: 사용자가 이겼습니다.")

# 위 경우가 아니면 무조건 컴퓨터가 이긴 거예요
else:
    print("결과: 컴퓨터가 이겼습니다.")
```

핵심 변경 포인트

while True — 무한 반복 시작

```
while True:
```

True는 항상 참이기 때문에 특별히 멈추라는 신호(**break**)가 없으면 계속 반복해요.

if user in choices — 목록 안에 있는지 확인

```
if user in choices:
    break
```

in은 “~안에 있나요?”를 확인하는 키워드예요.

```
"가위" in ["가위", "바위", "보"] → True ✓
```

```
"발차기" in ["가위", "바위", "보"] → False ✗
```

break — 반복 탈출

올바른 값이 들어오면 **break**로 **while** 반복문을 빠져나와요.

실행 예시

```
가위, 바위, 보 중 하나를 입력하세요: 발차기
"발차기"는 올바른 입력이 아니에요. 가위, 바위, 보 중 하나만 입력하세요!
가위, 바위, 보 중 하나를 입력하세요: 123
"123"는 올바른 입력이 아니에요. 가위, 바위, 보 중 하나만 입력하세요!
가위, 바위, 보 중 하나를 입력하세요: 바위
사용자 입력: 바위
컴퓨터 선택: 가위
결과: 사용자가 이겼습니다
```

흐름 요약: 잘못된 입력 → 경고 출력 → 다시 입력 요청 → 올바른 입력이 들어오면 **break**로 탈출 → 게임 진행!

▶ 활용해보자! – GOOD

점진적 업그레이드!

이 게임을 3판 2선승제로 바꿔줘.

```
수정된 전체 코드
import random # 랜덤 선택을 위한 모듈이 필요

choices = ["가위", "바위", "보"] # 선택지 목록이 필요

user_score = 0 # 사용자 점수를 저장해요 (처음엔 0점)
computer_score = 0 # 컴퓨터 점수를 저장해요 (처음엔 0점)

# 최대 3판을 진행해요
for round_num in range(1, 4): # 1, 2, 3 순서로 반복해요

    print(f"===== (round_num) 판 =====") # 몇 번째 판인지 보여줘요

    # 올바른 값을 입력할 때까지 계속 반복해요
    while True:
        user = input("가위, 바위, 보 중 하나를 입력하세요:")

        # if user in choices: # 올바른 입력이면 반복을 멈춰요
        #     break
        # else:
        #     print(f"({user})는 올바른 입력이 아니에요: 가위, 바위, 보 중 하나만 입력하세요!")

        computer = random.choice(choices) # 컴퓨터가 랜덤으로 선택해요

        print(f"사용자 입력: {user}")
        print(f"컴퓨터 선택: {computer}")

        # 이번 판의 승패를 확인해요
        if user == computer:
            print("이번 판: 무승부!")

        elif (user == "가위" and computer == "보") or #
            (user == "바위" and computer == "가위") or #
            (user == "보" and computer == "바위"):
            print("이번 판: 사용자 승리!")
            user_score += 1 # 사용자 점수를 1점 늘려요

        else:
            print("이번 판: 컴퓨터 승리!")
            computer_score += 1 # 컴퓨터 점수를 1점 늘려요

    # 현재 점수를 보여줘요
    print(f"현재 점수 → 사용자: {user_score} 컴퓨터: {computer_score}")

    # 2선승 확인: 누군가 2점에 먼저 도달하면 게임을 끝내요
    if user_score == 2:
        print(f"🎉 최종 결과: 사용자가 이겼습니다!")
        break # for 반복문을 바로 탈출해요

    if computer_score == 2:
        print(f"🎉 최종 결과: 컴퓨터가 이겼습니다!")
        break # for 반복문을 바로 탈출해요

# 3판을 다 했는데 아무도 2승을 못 하면 무승부예요
else:
    print(f"🎉 최종 결과: 무승부입니다!")
```

```
핵심 변경 포인트
점수 변수 2개 추가

user_score = 0
computer_score = 0

각 판이 끝날 때마다 이긴 쪽에 += 1로 점수를 더해요.

for 반복문으로 최대 3판 진행

for round_num in range(1, 4):

range(1, 4) 는 1, 2, 3 세 숫자를 순서대로 만들어줘요.

즉, 최대 3번 반복해요.

2점 도달 시 break 로 조기 종료

if user_score == 2:
    break

누군가 먼저 2점을 내면 남은 판은 건너뛰고 바로 게임이 끝나요.

for ~ else — 3판을 끝까지 다 한 경우

for ...:
    ...
else:
    print("무승부!")

for 의 else 는 break 없이 반복이 끝났을 때만 실행돼요.

즉, 3판을 모두 치렀는데 아무도 2승을 못 하면 무승부예요.
```

실행 예시

===== 1판 =====
가위, 바위, 보 중 하나를 입력하세요: 바위
사용자 입력: 바위
컴퓨터 선택: 가위
이번 판: 사용자 승!
현재 점수 → 사용자: 1점 컴퓨터: 0점

===== 2판 =====
가위, 바위, 보 중 하나를 입력하세요: 가위
사용자 입력: 가위
컴퓨터 선택: 바위
이번 판: 컴퓨터 승!
현재 점수 → 사용자: 1점 컴퓨터: 1점

===== 3판 =====
가위, 바위, 보 중 하나를 입력하세요: 보
사용자 입력: 보
컴퓨터 선택: 가위
이번 판: 컴퓨터 승!
현재 점수 → 사용자: 1점 컴퓨터: 2점

🎉 최종 결과: 컴퓨터가 이겼습니다!

▶ 활용해보자! – GOOD

오류 포착 → 디버깅!

3판 2선승제인데, 2승을 하지 않고 3판이 끝나니까 게임이 끝나

- Thought for 41s >
- 현재 코드를 확인하고 버그를 찾을게요.
- Read** c:\\$WAT\GoodExample\rps_streamlit.py (lines 51-70)
- 문제를 확인했어요. rps_streamlit.py:43-50의 게임 종료 조건이 잘못됐어요.
round_num == 3 이 되면 무조건 게임을 끝내고 있어서, 무승부 판이 꺼도 3라운드에서 끝나버려요. 2승이 나올 때까지만 게임을 종료하도록 조건을 수정할게요.
- Edit** c:\\$WAT\GoodExample\rps_streamlit.py
Removed 2 lines

```
04 # 게임 종료 조건 확인
05 if st.session_state.user_score == 2 or st.session_state.computer_score == 2:
06     st.session_state.game_over = True # 누군가 2점에 도달하면 끝!
07 elif st.session_state.round_num == 3:
08     st.session_state.game_over = True # 3판을 모두 마쳐도 끝!
09 else:
10     st.session_state.round_num += 1 # 아직 게임 중이면 다음 판으로 넘어가요

04 # 게임 종료 조건 확인! 2승을 먼저 달성한 쪽이 이겨요
05 if st.session_state.user_score == 2 or st.session_state.computer_score == 2:
06     st.session_state.game_over = True
07 else:
08     st.session_state.round_num += 1 # 아직 2승이 없으면 다음 판으로 넘어가요
```

수정 내용 요약

문제: round_num == 3 이면 무조건 게임 종료 → 무승부 판이 꺼면 2승 없이도 3라운드에서 끝남
수정 전 (잘못된 로직)

```
if user_score == 2 or computer_score == 2:
    game_over = True
elif round_num == 3: # ← 이게 문제! 2승 없어도 끝내버림
    game_over = True
else:
    round_num += 1
```

수정 후 (올바른 로직)

```
if user_score == 2 or computer_score == 2:
    game_over = True # 2승이 나왔을 때만 끝!
else:
    round_num += 1 # 아니면 계속 진행
```

이제 무승부가 몇 판이 나와도 반드시 2승을 달성한 쪽이 생길 때까지 게임이 계속돼요.

▶ 활용해보자! – GOOD

프론트엔드 구현

streamlit 화면으로 프론트까지 구현해줘

- Thought for 28s >
- GoodExample\rps_game.py 옆에 Streamlit 앱 파일을 생성할게요.

Write c:\\$STAT\GoodExample\rps_streamlit.py
147 lines

```
import streamlit as st  
import random
```

- GoodExample\rps_streamlit.py 파일이 생성됐어요!

실행 방법

먼저 Streamlit이 없으면 설치해요.

```
pip install streamlit
```

그 다음 실행해요.

```
streamlit run GoodExample\rps_streamlit.py
```

브라우저가 자동으로 열리면서 게임이 시작돼요.

주요 구현 포인트

항목	설명
<code>st.session_state</code>	버튼을 눌러도 점수 라운드가 사라지지 않게 저장해요
<code>st.button()</code>	가위·바위·보를 텍스트 입력 대신 버튼으로 선택해요
<code>st.rerun()</code>	버튼 클릭 후 화면을 즉시 새로 그려요
<code>st.balloons()</code>	이겼을 때 풍선 애니메이션이 날아올라요
<code>st.success / error / info</code>	결과에 따라 초록·빨강·파란 알림창을 보여줘요

▶ 그런데... AI는 완벽할까요?

?

“AI가 틀릴 수도
있을까요?”



완벽하다



아닐 수도 있다




위험1: AI가 자신 있게 틀립니다

AI는 틀린 정보도
자신 있게 말합니다

🔍 사실 확인 필수!

 AI CHAT


세종대왕이 맥북을 던진 사건에
대해 알려줘.

 거짓 정보

네, 조선왕조실록에 따르면
세종대왕은 훈민정음 창제 중
스트레스를 받아 맥북 프로를 던졌다고
기록되어 있습니다.
이 사건을 '맥북 투척 사건'이라고 부릅니다.



위험2: 개인정보 조심!

AI에 이름, 전화번호를 입력하면 어떻게 될까요?

절대 입력 금지!



이름 / 주민번호



전화번호



집 주소



비밀번호



입력한 정보가 AI 학습 데이터로
사용될 수 있습니다.



나의 소중한 개인정보가 다른 사람에게
유출될 위험이 있습니다.



실제 사례: 삼성전자

2023년, 삼성전자 엔지니어들이 ChatGPT를 통해
회사 기밀을 외부 서버로 유출시킴



위험3: AI 없이는 아무것도 못 해요?

생각하는 힘이 사라집니다

“AI 없이 숙제 못 하는 상황“

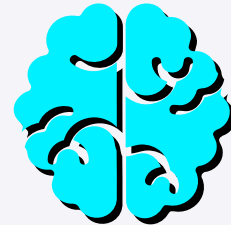
의존 🙄



AI에게 숙제를
다 맡기면

내 실력은 0

활용 💪

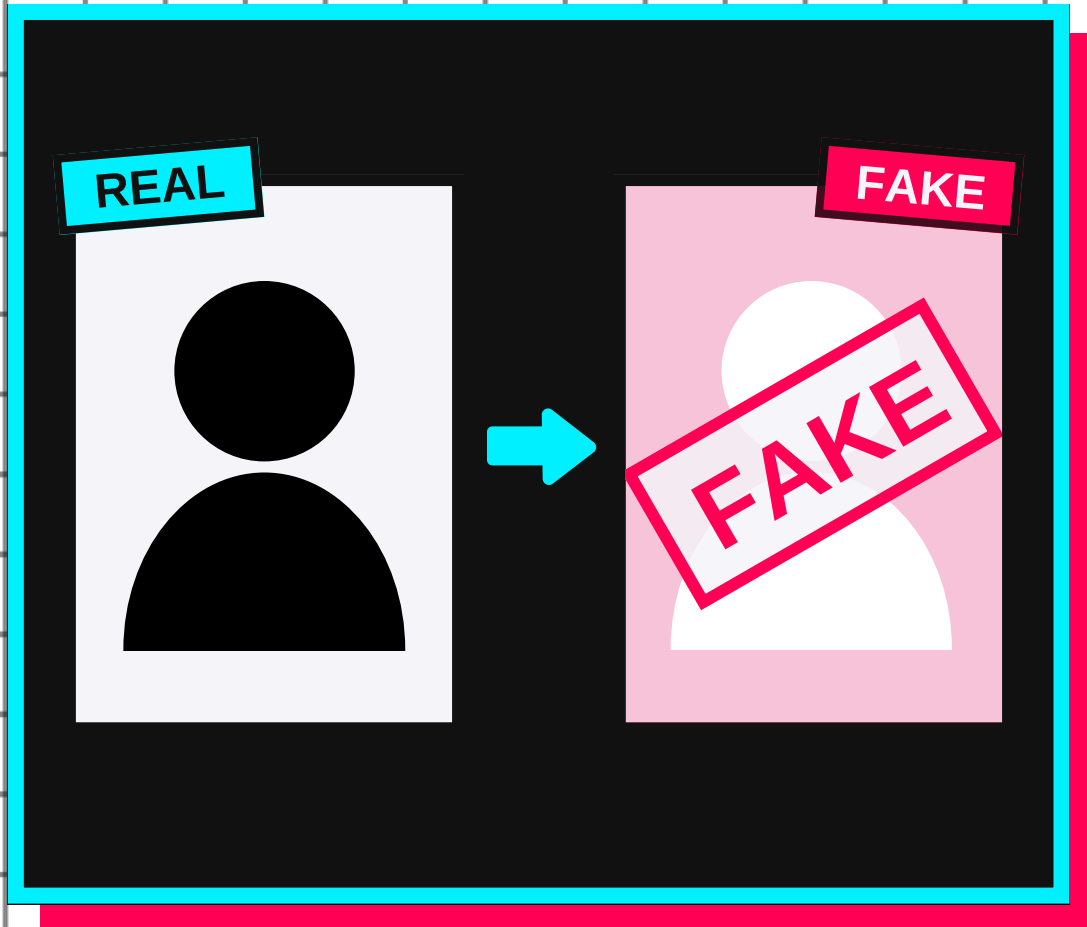


AI를 도구로
참고만 하면

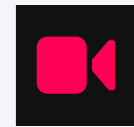
내 실력 UP!



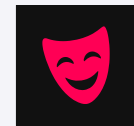
위험4: 딥페이크 - 가짜 영상



내 얼굴로 가짜 영상을
만들 수 있습니다



AI로 만든 가짜 영상/사진



유명인 사칭, 학교폭력 악용



법적 처벌 가능! 정보통신망법 위반

▶ 정리해볼까요?

AI = 강력한 도구

잘 쓰면 최고, 못 쓰면 위험

 잘 쓰면

 생산성 폭발

 창의력 확장

 시간 절약

 새로운 직업 창출

 못 쓰면

 가짜 정보에 속음

 개인정보 유출

 생각하는 힘 약화

 딥페이크 피해

VS

“AI를 두려워하지 말고, **활용하자!**”
“항상 **비판적으로** 생각하자!”

01



AI 답변은
항상 확인하기

02



개인정보는
절대 입력 금지

03



AI는 도구,
생각은 내가 하기

▶ Q&A

