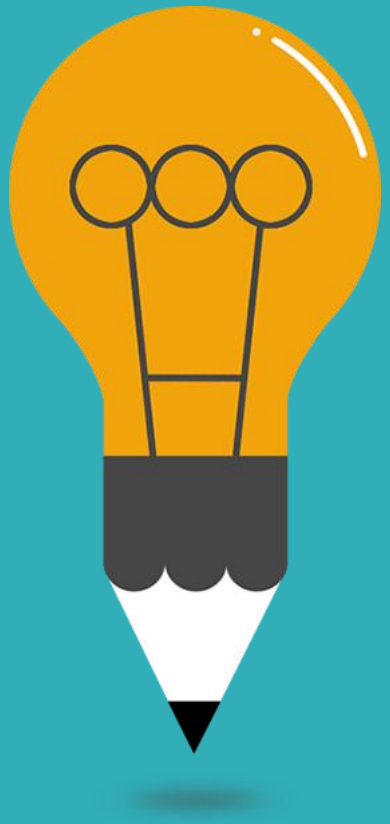
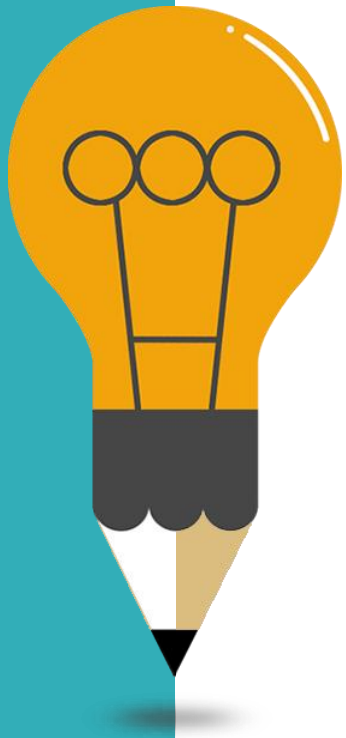


Season 1 온라인 코딩파티

도전! 인공지능 Tic-tac-toe

도전!
인공지능 Tic-tac-toe

수업 가이드



01 온라인 콘텐츠 개발 목표

02 온라인 콘텐츠 이용 대상

03 수업 연계 방안

04 단계별 수업 Tip

콘텐츠 개발 목표

프로그래밍, 학생들에게 가르치고 싶었던 것들

프로그래밍에 대한
긍정적인 인식

실생활 또는
다른 교과와의
연계

프로그래밍의
필요성과 가능성
이해

아이디어의
구현 능력

온라인 콘텐츠 개발 목표

> **프로그래밍의 여러 요소에 대한 개별화된 학습과 컴퓨팅 사고력을 Jump-Up할 기회**를 제공하며, 도전적인 과제를 해결해나가면서 얻은 **성취감과 즐거움**을 발판으로 더 발전된 학습을 해나갈 수 있도록 지원한다.

콘텐츠 이용 대상

교육과정 속 소프트웨어 교육

2015 개정 교육과정

초등학교	중학교	고등학교
Unplugged Activity	Block Coding	Text Coding
실과 SW 단원 17차시	정보 (필수) 34차시 이상	정보 (일반선택) 프로그래밍 등 (전문교과)

콘텐츠 이용 핵심 대상

고등학생

온라인 콘텐츠 이용 대상

“ 수업(정보 과목-문제해결과 프로그래밍)을 진행하면서 프로그래밍의 여러 요소와 컴퓨팅 사고력이 요구되는 **실제적인 프로그램 개발을 경험하고 싶은 학생** ”

수업 연계 방안

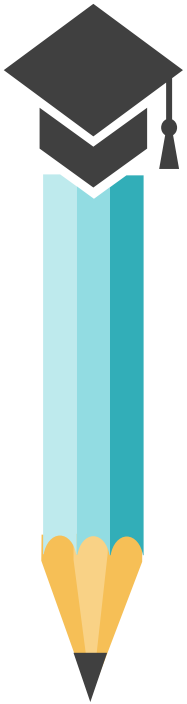
교육 목표

2015개정 고등학교 정보

고등학교 ‘정보’에서는 정보윤리의식을 바탕으로 정보 보호를 실천하기 위한 역량을 강화하고 실생활의 기초적인 문제뿐만 아니라 다양한 학문 분야의 복잡한 문제 해결을 위해 정보기술활용능력과 컴퓨팅 사고력, 협력적 문제해결력을 기르는 데 중점을 둔다.

성취 기준

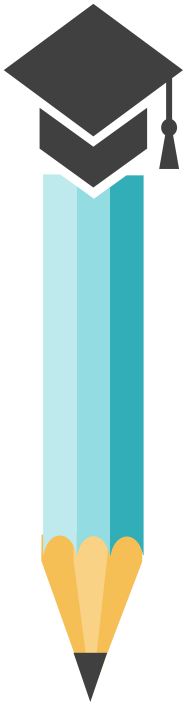
2015개정 고등학교 정보 – 추상화와 알고리즘



[12정보03-01]	복잡한 문제 상황에서 문제의 현재 상태, 목표 상태를 이해하고 목표 상태에 도달하기 위해 수행해야 할 작업을 분석한다.
[12정보03-02]	복잡한 문제 상황에서 문제 해결에 불필요한 요소를 제거하거나 필요한 요소를 추출한다.
[12정보03-03]	복잡하고 어려운 문제를 해결 가능한 작은 단위의 문제로 분해하고 모델링 한다.
[12정보03-04]	순차 구조, 선택 구조, 반복 구조 등의 제어 구조를 활용하여 논리적이고 효율적인 알고리즘을 설계한다.
[12정보03-05]	다양한 알고리즘의 성능을 수행시간의 관점에서 분석하고 비교한다.

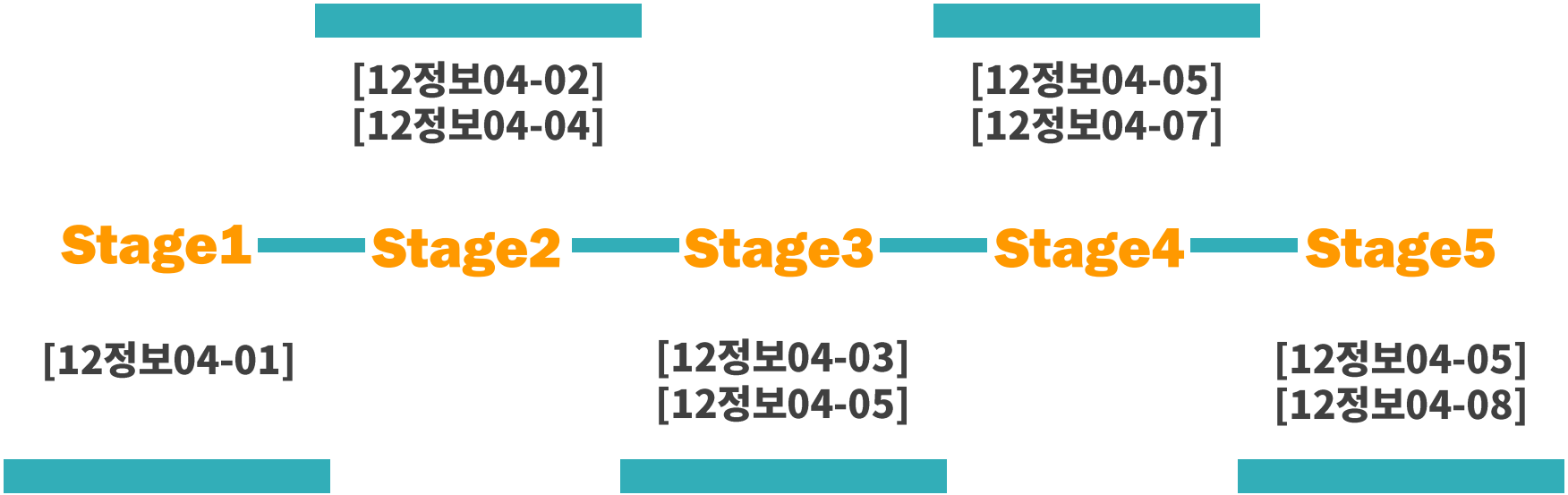
성취 기준

2015개정 고등학교 정보 – 프로그래밍



[12정보04-01]	텍스트 기반 프로그래밍 언어의 개발 환경 및 특성을 이해한다.
[12정보04-02]	자료형에 적합한 변수를 정의하고 이를 활용한 프로그램을 작성한다.
[12정보04-03]	다양한 연산자를 활용한 프로그램을 작성한다.
[12정보04-04]	표준입출력과 파일입출력을 활용한 프로그램을 작성한다.
[12정보04-05]	순차, 선택, 반복 구조를 활용한 프로그램을 작성한다.
[12정보04-06]	중첩 제어 구조를 활용한 프로그램을 작성한다.
[12정보04-07]	배열의 개념을 이해하고 배열을 활용한 프로그램을 작성한다.
[12정보04-08]	함수의 개념을 이해하고 함수를 활용한 프로그램을 작성한다.
[12정보04-09]	다양한 학문 분야의 문제 해결을 위한 알고리즘을 협력하여 설계한다.
[12정보04-10]	다양한 학문 분야의 문제 해결을 위해 설계한 알고리즘을 프로그램으로 구현하고 효율성을 비교·분석한다.

틱택토 콘텐츠와 성취기준



개발 콘텐츠와 성취기준

Stage6

[12정보04-03]
[12정보04-05]
[12정보04-06]

Stage7

[12정보04-03]
[12정보04-06]
[12정보04-08]

Stage8

[12정보04-05]
[12정보04-07]
[12정보04-08]

Stage9

[12정보04-02]
[12정보04-05]
[12정보04-07]

Stage10

[12정보03-03]
[12정보03-04]
[12정보04-02]

개발 콘텐츠와 성취기준

[12정보03-01]
[12정보03-02]
[12정보03-03]
[12정보03-04]

Stage11 — **Stage12** — **Stage13** — **성찰활동**

[12정보03-04]
[12정보04-05]
[12정보04-08]

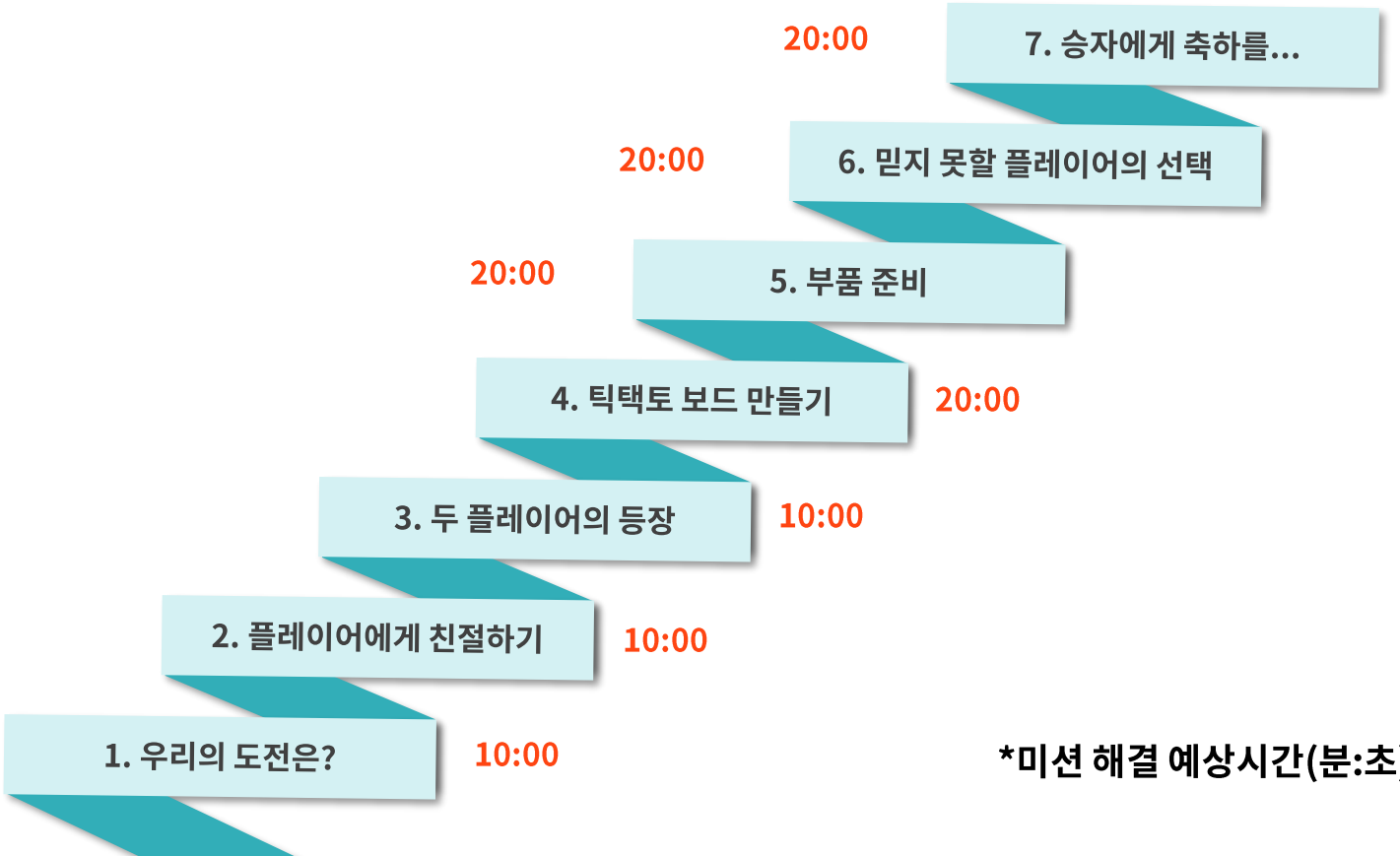
[12정보04-09]
[12정보04-10]

01 도전! 인공지능 Tic-tac-toe	난이도	학습내용	작성코드/전체코드(행)
Stage1 우리의 도전은?	★	코드 작성 및 실행 방법	1/1
Stage2 플레이어에게 친절하기	★	입출력문	4/12
Stage3 두 플레이어의 등장	★★	조건문과 반복문	3/26
Stage4 틱택토 보드 만들기	★★★	리스트	6/39
Stage5 부품 준비	★★★	함수	5/61
Stage6 믿지 못할 플레이어의 선택	★★★	다양한 연산자와 문자열 연산	4/70
Stage7 승자에게 축하를...	★★	조건문 및 함수 활용	5/96
Stage8 한 게임 더?	★★★	리스트 및 함수 활용	6/114
Stage9 컴퓨터와 대결	★★★	상수와 리스트 활용	6/127
Stage10 조금 똑똑해진 컴퓨터	★★★	변수(저장방식, 인수전달, 깊은복사)	7/140
Stage11 공정한 게임	★★	외부 모듈(random)	10/158
Stage12 나만의 인공지능 틱택토	★★★★★	게임 승리 전략 구상 및 구현	∞
Stage13 새로운 틱택토	★★★	게임 업그레이드	∞

수업 연계 방안

고등학교 정보 과목의 문제해결과 프로그래밍 영역의 모든 성취기준을 포함하고 있어 **정보 과목을 수강하는 학생이 개별적으로 이용하거나 교사가 수업 중 Jump-Up 과제로 이용**할 수 있다.

단계적 학습 및 게임 완성도 UP



*미션 해결 예상시간(분:초)

단계적 학습 및 게임 완성도 UP

*미션 해결 예상시간(분:초)

단계별 수업 Tip

온라인 콘텐츠(tic-tac-toe)

01 **우리의 도전은?**
도전 과제 및 실습환경 안내

02 **플레이어에게 친절하기**
입출력문과 변수(대입연산)을 이용한 기본 UI 개발

03 **두 플레이어의 등장**
제어문을 이용한 게임 진행 틀 개발

04 **틱택토 보드 만들기**
리스트를 활용한 게임 보드 표현

05 **부품 준비**
함수를 활용한 프로그램 구조화

06 **믿지 못할 플레이어의 선택**
상황 분석을 통한 입력 제어

07 **승자에게 축하를...**
승리할 조건 검사 및 게임 흐름 제어

08 **한 게임 더?**
제어문과 리스트를 활용해서 여러 게임 진행

온라인 콘텐츠(tic-tac-toe)

09 **컴퓨터와 대결**
플레이어와 컴퓨터가 대결하도록 변경

10 **조금 똑똑해진 컴퓨터**
컴퓨터의 기본 전략 추가

11 **공정한 게임**
random 모듈을 이용한 시작 순서 결정

12 **나만의 인공지능 틱택토**
컴퓨터의 승리 전략 추가

13 **새로운 틱택토**
게임 아이디어 추가

14 **성찰 활동**
활동 정리

Stage1
우리의 도전은?

도전 과제 및 실습 환경 안내

- 도전 과제로 정한 틱택토(tic-tac-toe) 게임의 규칙 안내
- 구름 환경에 코드를 입력하고 제출하는 과정 실습

Stage1. 수업 Tip(1)

게임 규칙에 대해 충분히 이해할 수 있도록 안내해주세요.

- 여러 스테이지를 진행한 후에도 무엇을 만들고 있는지 이해하지 못하는 경우가 있습니다.
- 게임을 여러 번 해보면서 이기기 위한 전략을 고민해보게 하면 [Stage12 인공지능 틱택토]에서 아이디어를 빨리 찾을 수 있습니다.

Stage1. 수업 Tip(2)

관찰과 시도가 프로그래밍을 배우는 빠른 길이라는 것을 알려주세요.

- 출력을 위해서 문자열은 숫자와 다르게 따옴표로 묶어야 되며, 함수의 인수는 소괄호로 둘러싸야 한다는 것 등을 꼼꼼히 살펴보고 테스트하고 작성해볼 수 있게 해주세요.

실습 환경의 이용 방법을 익히는 단계이기도 합니다.

- 매우 단순하지만 코드를 작성하고 오른쪽 위 파란색 버튼을 모두 살펴보도록 안내하면 이후 스테이지에서 편한 방법으로 테스트하고 제출할 수 있습니다.

Stage2
플레이어에게
친절하기

기본 유저 인터페이스 개발

- 게임 규칙에 대한 안내 출력(print 함수)
- 플레이어로부터 보드 위치 입력받아 형식에 맞게 출력 (input 함수, 변수와 대입연산자)

Stage2. 수업 Tip(1)

변수와 대입연산자

- 등호(=)는 대입연산자라는 점을 강조해주세요. 수학의 등호와 혼돈하는 경우가 많습니다. 대입연산자의 왼쪽에는 반드시 변수가 와야하고 오른쪽에는 변수에 대입할 값이 있어야 한다고 알려주시고 변수를 어떻게 사용하는지도 예시를 통해 설명해주세요.

변수의 이름 규칙

- 밑줄(_)과 문자로 시작해야 하며, 숫자가 포함될 수 있다는 기본 규칙을 여기서 알려주시면 좋겠어요.

Stage2. 수업 Tip(2)

함수 input

- 파이썬의 콘솔 입력이 다른 언어와 차이가 있어 다른 언어에 대한 경험이 있는 학생들은 혼란스러워하는 경우가 있습니다.
- 엔터키를 치기 전까지 입력되는 모든 내용을 하나의 문자열로 넘겨받는 함수라는 것을 다양한 입력으로 테스트하게 해서 이해시켜 주세요.
- input 함수의 실행 결과로 문자열을 넘겨받게 되므로 이후 입력받은 문자열을 이용하기 위해서는 대입연산자와 변수가 필요하다는 점도 알려주세요.

Stage2. 수업 Tip(3)

코드 작성

- 실행예시와 코드를 나란히 두고 비교하면서 코드를 완성하게 해주세요.
- 14행의 빈칸은 변수 inData와 pos 중 어느 것이든 상관없으며, 정수로 변환한 값이 대입된 pos는 이후 스테이지에서 사용됩니다. 위치가 문자가 아니라 숫자로 처리된다는 것을 알려주고 싶어 13행을 미리 넣어두었다고 보시면 됩니다.
- 코드를 복사할 때 블록을 선택하고 Ctrl+C를 눌러야 합니다. 오른쪽버튼을 클릭해서 단축메뉴를 열려고 하면 선택한 블록이 해제가 됩니다. 붙여넣기는 단축메뉴 사용 가능합니다.

Stage3
두 플레이어의
등장

게임 진행 틀 구현

- 제어문(조건문, 반복문)
- 두 플레이어가 번갈아가며 보드의 위치를 선택
- 9번 입력받고 프로그램 종료

Stage3. 수업 Tip(1)

제어문

- 반복문(for, while)과 조건문, 비교연산자까지 포함되어 있어 내용이 많습니다.
- 학생들이 지치지 않도록 예제를 통해서 의미를 파악하는데 집중할 수 있도록 지도해주세요.
- 실습환경(구름)에서 들여쓰기가 같아 보여도 Tab과 Spacebar로 들여쓰기한 것을 구분하니 주의하도록 안내해주세요. 실행시 들여쓰기 오류 (Indentation Error 등)이 발생하면 Tab과 Splacebar를 혼용한 것이 아닌지 먼저 확인해보세요.

Stage3. 수업 Tip(2)

코드 작성

- 대부분의 스테이지에서 기본 코드를 제공해줬습니다. 기본 코드를 주고 완성하게 하는 것이 적절한 속도로 진행하기 위한 것이기도 하지만, 코드를 읽는 능력과 좀더 정제된 코드로 표현할 수 있게 안내하는 역할도 있다고 봅니다.
- 이미 작성되어 있는 코드를 파악하는 것부터 시작하도록 지도해주세요.
- 실행결과와 비교하면서 주석을 참고해서 작성해야 합니다.

Stage3. 수업 Tip(3)

코드 작성

- 17행은 돌을 둔 횟수를 세는 구문입니다. 위치를 입력받은 후 numOfStone을 1씩 증가시켜주면 됩니다.
- 19행에서 돌을 둔 횟수가 9이상인지 검사하는 것은 9번만 입력받고 중단하기 위해서이니, 20행은 반복문을 빠져나갈 수 있도록 **break** 구문이 필요합니다.
- 23행은 번갈아가며 돌을 둘 수 있도록 currentStone가 'O'인 경우, 즉 **currentStone == 'O'**를 조건으로 주어야 합니다. 숫자 '0'를 대문자 'O' 대신 잘못 쓰는 경우가 종종 있으니 주의가 필요합니다.

Stage4
틱택토 보드
만들기

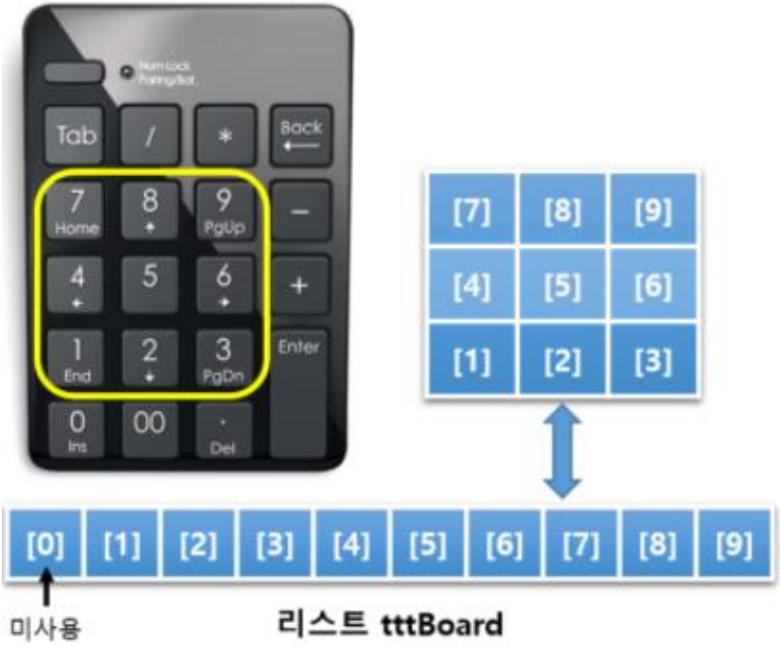
게임 보드 표현

- 리스트를 활용해서 틱택토 보드 표현
- 보드의 위치를 키보드의 숫자 키패드로 선택
- 선택한 위치의 보드에 돌('O' 또는 'X') 표시
- 보드의 상태 출력

Stage4. 수업 Tip(1)

리스트

- 틱택토 보드의 각 칸의 상태를 표현하기 위해 9개의 변수를 사용한다면 반복문을 이용할 수 없고 구현이 복잡해지기 때문에 여러 개의 항목을 가지는 리스트 구조가 필요하다는 점을 알려주세요.
- 키보드의 숫자 키패드를 틱택토 보드와 매칭해서 입력에 이용하고 입력된 숫자를 리스트의 인덱스와 매칭하려는 의도를 파악하도록 도와주세요.



Stage4. 수업 Tip(2)

코드 작성

- 29행은 입력받은 위치(pos)에 현재 플레이어의 돌(currentStone)을 두는 과정입니다.
- 리스트 tttBoard의 인덱스와 입력받은 위치(pos)가 동일한 의미로 사용되므로 아래와 같이 구문을 작성해주면 됩니다.
- **tttBoard[pos] = currentStone**

Stage4. 수업 Tip(3)

코드 작성

- 실행예시를 통해 게임 시작 전과 플레이어의 입력을 받을 때마다 보드의 현재 상태가 출력된다는 것을 확인하도록 해주세요.
- 게임 시작 전, 즉 반복문 시작 전에 보드의 상태를 출력한 구문을 그대로 이용하면 됩니다. 단, 들여쓰기에 주의하도록 해주세요. 꼭 Tab을 이용해서 들여쓰기 하도록 가이드에도 문구를 넣어뒀으나 놓치는 학생이 많습니다.
- 코드를 복사할 때는 Ctrl+C를 이용해주세요.

Stage5
부품 준비

프로그램 구조화

- 함수 정의
  - printInfo: 게임 안내문 출력
  - drawBoard: 보드 상태 출력
  - numberOfStone: 보드에 놓여진 돌의 개수 리턴
  - ticTacToe: 틱택토 1게임 진행
- 정의된 함수를 활용해서 구조적인 프로그램 작성

Stage5. 수업 Tip(1)

함수

- 작년 수업에서 함수를 배우기 전에 이 콘텐츠를 사용한 학생들이 가장 어려워했던 단계입니다.
- 함수를 처음 접하는 학생들이라면 더 자세한 함수에 대한 설명보다는 코드를 완성하기 위해 필요한 설명(함수 호출 등)으로 다른 단계보다는 조금 더 도움을 주면 좋겠습니다.

Stage5. 수업 Tip(2)

코드 작성

- 각 함수의 기능과 매개변수, 리턴값을 파악하도록 도와주세요.(주석 참고)
- printInfo 함수 정의(헤더): 함수 호출문(40행)에 인수가 하나도 없으므로 함수 헤더는 **def printInfo():** 라고 작성해야 합니다.
- numberOfStone 함수 정의(내부 블록)
  - 틱택토 보드(리스트)에 놓인 돌의 개수를 계산하고 리턴해야 합니다.
  - 리스트의 모든 항목을 검사해서 빈칸('-')이 아닌 항목의 개수를 세도록 27행에 조건문 **if c != '-':** 을 넣습니다.
  - 29행에 계산한 돌의 개수를 리턴하는 구문 **return n** 이 필요합니다.

Stage5. 수업 Tip(3)

코드 작성

- drawBoard 함수 호출: 보드의 상태를 출력하는 함수는 게임 시작 전(반복문 이전)에 한 번 호출되고, 돌의 위치를 입력받을 때마다 호출되어야 합니다. 49행은 41행과 똑같이 drawBoard 함수를 호출하면 됩니다.
- ticTacToe 함수 호출: 틱택토 게임 1번을 진행하는 함수인 ticTacToe(33행~57행)는 정의만 되어 있으므로, 호출을 해야 함수가 실행됩니다. 함수 밖 61행에 ticTacToe() 라고 작성하면 함수가 실행됩니다.

Stage6
믿지 못할
플레이어의 선택

상황 분석을 통한 예외 처리

- 플레이어의 입력 상황 분석
- 잘못된 입력에 대한 예외 처리
  - ‘1’에서 ‘9’ 사이의 숫자가 아닌 경우
  - 선택 위치가 비어있지 않은 경우

Stage6. 수업 Tip(1)

연산자

- 산술, 비교, 논리, 문자열 연산자를 안내하고, 예시를 제공하고 있습니다.
- 다른 연산에 비해 논리연산을 어려워하므로 가능하다면 논리연산의 예시를 조금 더 소개해주세요. 논리연산자의 피연산자는 논리값(True, False)이어야 하며 일반적인 숫자나 문자열도 논리연산에서 사용되면 논리값의 의미를 갖게 됩니다. 아래는 우리가 배운 자료형 중 조건식에서 False의 의미를 갖는 값이며, 그 외의 값은 True의 의미를 갖습니다.
  - 수치형 데이터의 0 또는 0.0
  - 빈 시퀀스 자료형: “”(빈 문자열), [](빈 리스트)

Stage6. 수업 Tip(2)

연산자

- 학생들이 많이 실수하는 경우입니다.
- if c == ‘A’ or ‘B’:
- 변수 c가 ‘A’이거나 ‘B’인지 확인하는 조건문을 작성하려는 의도인데 조건식의 결과는 항상 True입니다. 아래와 같이 작성해야 의도한대로 실행됩니다.
- if c == ‘A’ or c == ‘B’:
- 또는 리스트의 in 연산자를 배운 경우 이렇게 작성해도 됩니다.
- if c in [‘A’, ‘B’]:

Stage6. 수업 Tip(3)

코드 작성

- **getPlayerMove 함수 정의**
  - **‘1’부터 ‘9’ 사이의 문자가 입력되었는지 검사하는 조건식(27행)은 [배우기]의 마지막 예시를 참고하면 쉽게 해결할 수 있습니다.**
    - **len(inData) == 1 and ' 1 ' <= inData <= ' 9 '**
  - **28행은 정수로 변환하는 함수 int를 사용하면 됩니다.**
  - **29행은 리스트 중 입력받은 위치가 비어있는지(39행의 조건식을 참고) 확인하는 조건식이 필요합니다.**
    - **board[pos] == ‘-’**

Stage6. 수업 Tip(4)

코드 작성

- **getPlayerMove 함수 호출(56행)**
  - **23행 주석과 24행 함수의 헤더부분을 참고해서 첫 번째 인수는 틱택토보드 상태를 저장하는 리스트, 두 번째 인수는 누구 차례인지 나타내는 변수가 필요하다는 것을 파악하게 해주세요.**
    - **getPlayerMove(tttBoard, currentStone)**

Stage7
승자에게
축하를...

게임 흐름 제어

- 승리할 조건 검사
- 게임 종료 조건 추가
  - 이긴 플레이어가 있는 경우
  - 보드가 꽉 찬 경우(비긴 경우)

Stage7. 수업 Tip(1)

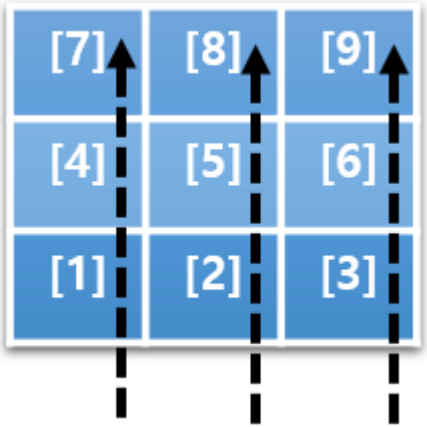
함수 응용하기

- return 구문은 함수에서 2가지 기능이 있습니다. 첫 번째는 함수의 결과값을 호출한 곳으로 넘겨주는 것이고, 두 번째는 함수의 실행을 끝내고 제어를 호출한 곳으로 넘기는 것입니다.
- 일반적으로 return 구문을 함수의 마지막에 두지만 함수를 끝까지 실행하지 않고 제어를 호출한 곳으로 넘겨야 할 경우가 종종 있습니다. 이번 스테이지의 미션에서도 이것을 적용하면 간결하게 함수를 완성할 수 있습니다.

Stage7. 수업 Tip(2)

코드 작성

- **isWinner 함수 정의**
  - **세로 열 중 플레이어의 돌로 채워진 열이 있는지 확인하는 코드는 바로 위 가로 열을 확인하는 코드를 참고해서 작성하되 규칙을 잘 찾아보도록 지도해주세요.**




```
for i in [1, 2, 3]:
    if board[i] == stone and board[i + 3] == stone and board[i + 6] == stone:
        return True
```

Stage7. 수업 Tip(3)

코드 작성

- **isWinner 함수 정의**
  - **대각선(좌상, 우하)을 플레이어의 돌로 채웠는지 확인하는 코드는 바로 위 코드를 참고해서 하면 됩니다.**

```
if board[3] == stone and board[5] == stone and board[7] == stone:
    return True
```

  - **마지막에 리턴할 값은 False입니다. 완성된 열이 있다면 이미 True를 리턴하고 함수가 종료되기 때문에 이 코드가 실행된다면 완성된 열이 없습니다.**

Stage7. 수업 Tip(4)

코드 작성

- ticTacToe 함수 정의
  - isWinner 함수에서 True를 리턴받은 경우(승자가 있는 경우) 누가 이겼는지 출력하는 구문에서 빈칸에는 현재 플레이어의 돌(currentStone)을 출력하면 됩니다. 현재 플레이어가 둔 후에는 현재 플레이어가 이겼는지 확인하기 때문에 승자는 현재 플레이어가 되겠지요.
  - 비긴 경우를 검사하는 조건식은 보드가 꽉 찾는지 확인하면 됩니다.
    - numberOfStone(tttBoard) >= 9

Stage8
한 게임 더?

여러 게임을 진행할 수 있도록 변경

- 제어문을 활용해서 여러 게임 진행
- 각 게임의 결과를 저장(리스트)해두었다가 마지막에 출력

Stage8. 수업 Tip(1)

함수 복습

- 이전 스테이지의 함수에 대한 설명과 중복되는 부분입니다.
- Stage8의 미션은 게임을 여러 번 진행하고 그 결과를 마지막에 정리해서 출력하는 것입니다.
- 1게임을 진행하는 ticTacToe 함수에서 승자가 결정되거나 보드가 꽉 찬 경우 바로 게임 결과를 리턴하게 하고, 호출한 곳에서는 이 결과를 저장해두어야 합니다.
- 함수의 리턴값에 대해 한 번 더 실습해보게 됩니다.

Stage8. 수업 Tip(2)

리스트 심화

- 틱택토 보드의 상태를 저장하기 위해 리스트를 사용하고 있습니다. 이번 스테이지에서는 빈 리스트를 생성하고 리스트에 항목을 추가하는 방법을 배웁니다.
- 리스트는 항목의 수정, 삽입, 삭제가 모두 가능한 변경 가능한 데이터구조라는 점을 알려주시고, 삽입하는 방법을 예시를 통해 확인하도록 지도해주세요.
- 리스트의 append 메소드에 대해서 관찰을 통해 사용방법만 익히도록 해주세요. 클래스나 메소드에 대한 이해는 이 콘텐츠의 범위를 벗어납니다.

Stage8. 수업 Tip(3)

코드 작성

- **ticTacToe 함수 정의**
  - **실습내용 2의 설명에 따라 85행에는 승자가 누구인지 리턴해야하므로 빈 칸에는 currentStone을 넣으면 됩니다.**

Stage8. 수업 Tip(4)

코드 작성

- **메인 프로그램 완성(98행~)**
  - **반복문을 이용해서 게임을 여러 번 진행할 수 있도록 하고, 게임의 결과를 리스트에 추가해둡니다.**

```
resultList = []
while True:
    result = ticTacToe()
    resultList.append(result)

    isContinued = input("한 게임 더 진행할까요?(y/n) ")
    if isContinued == 'n':
        break
```

Stage8. 수업 Tip(5)

코드 작성

- **메인 프로그램 완성(98행~)**
  - **게임 결과가 저장된 resultList의 값을 출력합니다. ticTacToe 함수에서 비길 경우 '-'을 리턴했으므로 비긴 경우를 확인하는 구문은 아래와 같습니다.**

```
for i in range(len(resultList)):
    print(i+1, ':', end = ' ')
    if resultList[i] == '-':
        print("비김")
    else:
        print(resultList[i], '승')
```

Stage9
컴퓨터와 대결

컴퓨터와 게임을 진행하도록 변경

- 플레이어와 컴퓨터가 번갈아 보드 위치 선택
- 컴퓨터는 처음 찾은 빈칸에 둠(다음 Stage에서 개선)

Stage9. 수업 Tip(1)

Computer-Player 버전

- 이번 스테이지에서는 컴퓨터와 플레이어가 대결하도록 코드를 변경합니다.
- 2 player 버전보다 더 고민해야 될 부분이 많아집니다. 플레이어가 두는 위치는 입력만 받으면 되지만, 컴퓨터는 우리가 두는 위치를 지정해주어야 합니다.

Stage9. 수업 Tip(2)

상수화된 변수

- 파이썬은 상수 변수를 지원하지 않지만, 전체가 대문자로 된 변수는 일반적으로 한 번 값을 지정하고 프로그램에서 값을 변경하지 않는 상수형 변수로 사용합니다.
- 이번 스테이지에서는 플레이어 돌은 'O', 컴퓨터 돌은 'X'로 프로그램을 시작할 때 결정하고 변경하지 않기 때문에 직관적으로 이해하기 쉽게 PLAYER_STONE, COMPUTER_STONE이라는 변수에 저장해서 사용합니다.

Stage9. 수업 Tip(3)

리스트의 항목 개수 세기

- 리스트에 특정 항목이 몇 개 들어있는지 셀 수 있는 메소드 count를 예시를 통해 배울 수 있습니다.
- 여러 게임 진행 후에 플레이어의 입장에서 “O승 O패 O무”로 결과를 출력할 때 이용합니다.

Stage9. 수업 Tip(4)

코드 작성

- **getComputerMove 함수 정의**
  - **컴퓨터가 보드에 돌을 두도록 하는 함수입니다. 아직 컴퓨터는 똑똑하지 못해 앞에서부터 순서대로 빈자리를 찾아 둡니다.**

```
for i in range(1, 10):
    if board[i] == '-':
        board[i] = stone
            return #또는 break
```

Stage9. 수업 Tip(5)

코드 작성

- **printInfo 함수 호출**
  - **프로그램 시작 시점에 한 번만 게임 안내를 출력하기 위해서는 함수 밖 메인 영역으로 이동해야 합니다. 이동 가능한 곳이 여러 곳 있지만 메인 프로그램이 시작되는 119행으로 이동하는 것을 추천합니다.**

Stage9. 수업 Tip(6)

코드 작성

- **ticTacToe 함수 정의**
  - **플레이어가 먼저 두는 것으로 정했으므로 반복문 안에서 플레이어가 먼저 두고 컴퓨터가 두게 합니다. 컴퓨터가 두는 과정은 플레이어가 두는 코드를 참고하면 대부분의 학생들이 쉽게 완성합니다.**

```
getComputerMove(tttBoard, COMPUTER_STONE)
drawBoard(tttBoard)

if isWinner(tttBoard, COMPUTER_STONE):
    print("아쉽군요. 컴퓨터가 이겼습니다.")
    return COMPUTER_STONE
```

Stage9. 수업 Tip(7)

코드 작성

- **메인 프로그램 완성(131행)**
  - **리스트의 count 메소드를 이용해서 코드를 완성할 수 있습니다.**

```
print(resultList.count(PLAYER_STONE), '승 ',
      resultList.count('-'), '무 ',
      resultList.count(COMPUTER_STONE), '패', sep = '')
```

Stage10
조금 똑똑해진
컴퓨터

컴퓨터의 게임 전략 추가

- 컴퓨터의 전략(순서)
    1. 컴퓨터가 두면 이길 위치를 찾아 둠
    2. 플레이어가 다음에 두면 이길 위치를 막아줌
    3. 빈 곳을 찾아 둠

Stage10. 수업 Tip(1)

기본 전략을 가지고 컴퓨터가 위치를 선택하도록 업그레이드

- 기본 전략은 간단합니다. 컴퓨터가 두면 이길 위치가 있으면 그 위치에 두어 승리합니다. 그렇지 않고 다음 차례에서 플레이어가 두면 이길 위치가 있으면 그 위치에 두어서 플레이어의 승리를 막아 줍니다.
- 두 경우 모두 해당이 없으면 이전 스테이지처럼 앞에서부터 빈자리에 둡니다.

Stage10. 수업 Tip(2)

파이썬 변수의 대입 방식

- 예제를 통해 파이썬 변수가 하나의 메모리 공간을 지속적으로 가리키는 것이 아니라는 것을 확인하게 합니다. 다른 값이 대입되면 값이 저장된 공간을 가리키게 된다는 것을 이해하면 함수의 호출시 넘긴 인수와 매개변수 사이의 관계도 쉽게 파악할 수 있습니다.

파이썬 리스트 변수의 대입 방식

- 리스트 내부의 항목이 변경된다고 해서 일반 변수처럼 다른 공간을 가리키지는 않습니다. 따라서 원래 리스트를 변경하고 싶지 않다면 깊은 복사가 필요하다는 것을 설명해주세요.

Stage10. 수업 Tip(3)

코드 작성

- **getComputerMove 함수 정의**
  - **컴퓨터가 두면 이길 위치를 찾는 코드를 참고해서 플레이어가 두면 이길 위치를 찾아 컴퓨터의 돌을 두는 코드를 완성합니다.**

```
for i in range(1, 10):
    if board[i] == '-':
        copyBoard = board[:]
        copyBoard[i] = PLAYER_STONE
        if isWinner(copyBoard, PLAYER_STONE):
            board[i] = COMPUTER_STONE
            return
```

Stage11
공정한 게임

게임 시작 순서를 랜덤하게 결정

- random 모듈
- 컴퓨터와 플레이어 중 누가 먼저 시작할지 랜덤하게 결정

Stage11. 수업 Tip(1)

누가 먼저 둘지 랜덤하게 결정하도록 업그레이드

- 랜덤하게 순서를 정하기 위해 random 모듈을 이용합니다.
- ticTacToe 함수의 일부 구조가 변경되므로 먼저 제공된 코드를 분석해보도록 지도해주세요.
- 랜덤하게 진행되므로 이번 스테이지부터 자동채점이 지원되지 않습니다.

Stage11. 수업 Tip(2)

모듈 이용(random)

- 파이썬은 다양한 기능을 모듈화해서 제공하고 있다는 것을 알려주세요.
- 터틀을 배운 적인 있다면, turtle 모듈은 프로그래밍 학습을 위해 선이나 도형을 쉽게 그릴 수 있는 기능을 가진 모듈이라는 것을 알려주시고, 그 외에도 데이터분석을 위해 사용하는 numpy, pandas, pyplotlib나 AI를 위해 사용하는 tensorflow, 게임 제작을 위한 pygame 등 많은 모듈이 있다는 것도 알려주세요.
- radom 모듈 중 이번 스테이지에서는 choice 메소드를 활용합니다.

Stage11. 수업 Tip(3)

코드 작성

- **import 모듈 가져오기(1행)**

```
import random
```

- **두 문자 중 하나를 랜덤하게 선택하는 구문(31행)**

```
first = random.choice(['O', 'X'])
```

- **whoIsFirst 함수 호출(123행)**

```
currentStone = whoIsFirst()
```

Stage11. 수업 Tip(4)

코드 작성

- **컴퓨터가 두는 경우 작성(136행~)**

```
else:
    getComputerMove(tttBoard, COMPUTER_STONE)
    drawBoard(tttBoard)

    if isWinner(tttBoard, COMPUTER_STONE):
        print("아쉽군요. 컴퓨터가 이겼습니다.")
        return COMPUTER_STONE

    currentStone = PLAYER_STONE
```

Stage12
나만의
인공지능 틱택토

컴퓨터의 승리 전략 추가

- 컴퓨터의 승리 전략을 최대한 많이 추가
- 테스트와 보완을 통해 개선

Stage12. 수업 Tip

컴퓨터가 필승 전략을 가지도록 업그레이드

- 알려진 알고리즘을 이용해서 작성하는 것을 요구하지 마세요.
- 예를 들어, 오목의 삼삼 경우처럼 두 줄에 두 개씩 돌이 놓일 수 있는 위치를 찾아 두는 등의 전략을 추가할 수 있습니다.
- 매우 단순한 게임이기 때문에 친구와 함께 테스트하며 컴퓨터가 지는 경우를 찾아 개선하면서 즐겁게 완성도를 높여갈 수 있도록 안내해주세요.
- 프로그래밍 수업에 관심이 없던 학생 중 테스트를 아주 잘해 친구들 프로그램의 허점을 찾아주면서 수업에 함께 참여한 학생이 있었습니다. 코딩 능력만큼 검증 능력도 중요하다고 칭찬해주시면 많은 학생들이 참여할 수 있을 거라고 기대합니다.

Stage13
새로운 틱택토

게임 아이디어 추가

- 틱택토 게임을 더 재미있게 만들어줄 규칙 추가
- 새로운 게임 아이디어 제안

Stage13. 수업 Tip

게임 요소 추가 또는 새로운 게임 아이디어 제안

- 구현에 초점을 맞추면 학생들이 너무 부담을 가질 수 있을 거라고 생각합니다.
- 게임에 대한 새로운 아이디어를 제안하고 서로 공유하면서 발전할 수 있는 기회를 마련해주세요.

감사합니다.