공학도를위한창의적컴퓨팅 과제#4

~ 1. 게임 설명 ~

도입부

때는 적당히 먼 미래, 단거리 워프 기술과 유기화합물 합성기 소형화 기술, 그리고 강화된 개인정보보호법을 준수하는 개인화 다중 평면 교차 기술(약칭 PMPCT)이 널리 쓰이는 시대입니다. 긴 동면에서 깨어나 새로운 땅에 도착한 여러분은, 인류의 터전을 넓힌다는 기본 목표를 달성하기 위해 지금부터 무엇을 할 지 고민하기 시작합니다. 약 세 시간 뒤면 저녁밥을 먹어야 하니, 그 전까지, 효율성을 챙겨 가며, 동료들과 함께, 간만에 힘 좀 써 보기로 마음먹고 발걸음을 옮깁니다.

캐릭터와 세계

프로그래머인 내가 할 일: 캐릭터 하나 만들기

과제 참여자 여러분은 영토 확장을 위해 행동하는 캐릭터들 중 하나(이하 '내 캐릭터')를 만들게 됩니다. 구체적으로, 내 캐릭터의 이름과 능력치를 정하는 **수식**, 그리고 '이제 뭐 하지'에 대한 의사 결정을 수행하기 위한 **함수**에 대한 **함수 정의**를 적어 둔 **모듈**(.py 파일)을 제출하게 됩니다. 이전 과제들에 비해 훨씬 다양한 Data를 다루어야 하니, 일단 각종 설명서들 및 영상들을 충분히 구경해 본 다음 '그래서 내 캐릭터는 어떤 스타일로 만들 것인지'를 생각해 보면 좋겠습니다.

마감 시간이 지나면 여러분이 만든 캐릭터들을 모아 평면 위에 올려 놓고 총 10,000초 동안 '본 게임'을 진행합니다. 진행 도중에 다양한 부문들에 대한 점수를 집계하게 되며, 각 스타일에 대해 마련해 둔 조건을 하나 이상 만족하면 과제 만점으로 간주됩니다. 부문별 가장 높은 점수를 낸 캐릭터(를 만든 친구)에게는 작은 상품을 드릴 예정이니, 시험 공부나 후반부 프로젝트 연습을 겸하여 좀 더 높은 효율을 내기 위해 노력해 봅시다.

다중 평면 세계

이 게임의 세계는 여러 평면들의 조합으로 구성되어 있습니다. 구체적으로, 모든 캐릭터들이 볼 수 있으며 칸을 점유하거나 각종 시설을 건설할 수 있는 공유 평면 하나와, 캐릭터마다 고유한, 점유 또는 건설에 사용하기 위해 수집할 수 있는 자원들이 놓여 있는 개인 평면들이 있습니다. 모든 평면은 교차되어 있으며, 캐릭터들은 공유 평면과 자신의 개인 평면에 '동시에' 존재합니다(강화된 개인정보보호법에 의거, 다른 캐릭터의 개인 평면을 볼 수는 없어요). 게임의 기본 목표는 공유 평면 위의 칸을 최대한 많이 점유하는 것으로, 이를 위해 각 캐릭터들은 자신의 개인 평면에서 자원을 수집하여 직접 작업에 활용하거나, 다른 캐릭터들이 들고 갈 수 있도록 공유 평면 위 거점에 수납해 둘 수 있습니다. 물론, 직접 자원을 수집하지 않고 다른 캐릭터들이 모아 놓은 자원을 인출하여 작업에 활용하는 것 또한 가능합니다. 이러한 '자원 흐름'의 어느 부분을 주로 수행할 것인지를 결정하는 것이 내 캐릭터의 '스타일'을 정하는 첫 단계가 된다 보면 되겠습니다.

캐릭터들이 이 세계에서 하는 일: 서로 도우며 자원 모으고 사용하기

이 게임은 개인 평면 위 자원을 집어 공유 평면에 놓는 게임이에요

막 동면에서 깨어난 캐릭터들에게 제시된 기본 목표는 제한 시간 안에 공유 평면 위의 칸들을 최대한 많이 점유하는 것입니다. 그러나 점유에는 자원이 꽤 많이 필요하며, 정작 공유 평면에는 자원이 단 하나도 놓여 있지 않습니다. 그러므로, 누군가는 자신의 개인 평면에서 자원을 수집해 올 필요가 있습니다. 수집 행동을 자주 선택하는 캐릭터가 많을수록, 공유 평면에서 사용 가능한 자원의 양을 그만큼 다량 확보할 수 있다고 말할 수 있습니다.

수집가 스타일과 작업자 스타일, 그리고 거점

여기서, 각 캐릭터가 한 번에 들 수 있는 자원의 양에 한계가 있는데다 이 게임에는 기회 비용 도입을 위한 속칭 '후딜레이' 및 '기력' 시스템이 존재하기 때문에, 점유에 사용할 자원을 그 근처 칸에서 직접 수집하여 조달하는 것(또는 관점에 따라, 수집한 자원을 직접 다른 칸까지 들고 가서 점유에 사용하는 것)은 그리 효율적이지 않습니다. 대신, 캐릭터들은 공유 평면 위에 있는 거점에 들고 있는 자원을 수납하거나, 거점에 있는 자원을 인출할 수 있습니다. 그렇다 보니 거점이 있는 칸은 드넓은 평면 위에서 수집을 중시하는 캐릭터들(이하 '수집가' 스타일로 지칭)과 점유 등 작업을 중시하는 캐릭터들(이하 '작업자' 스타일로 지칭)이 들러 자원 흐름을 발생시키는, 일종의 협력 지점으로 작용하게 됩니다.

거점과 확장

거점은 캐릭터들이 게임을 시작할 때 위치하는 칸(좌표 (0, 0), 이하 '중심' 칸)에 있으며, 자원을 사용하여 다른 칸에 새 거점을 건설할 수도 있습니다. 당연하게도 거점 건설은 칸 점유보다 훨씬 많은 자원을 필요로 합니다. 그러나 일단 거점을 한 번 건설해 두면 수집가 캐릭터들이 그 거점 주변에서 더 효율적으로 자원을 수집, 수납할 수 있으며, 거점에 수납되어 있는 자원이 많을수록 작업자 캐릭터들이 보다 수월하게 점유 또는 건설을 위한 자원을 확보할 수 있게 됩니다. 따라서 이전보다 더 멀리 있는 칸으로 확장하기 위해서는 적당한 시점에, 적당한 자리에 거점을 건설해 둘 필요가 있을 것입니다.

텔레포터 네트워크

이동, 특히 자원을 든 채 이동하는 것은 시간 및 기력 소모 측면에서 많은 비용이 듭니다. 모든 캐릭터들이 중심 칸 주변에 모여 있는 초반에는 몇 칸만 이동해도 다른 거점으로 향할 수 있지만, 어느 정도 확장이 진행된 이후에는 각 거점들 사이를 직접 이동하는 것이 사실상 불가능해질 수 있습니다. 다행히도 이 게임 속 세계에는 단거리 워프 기술이 상용화되어 있어, 텔레포터를 건설해 둔 칸들 사이를 그 거리와 무관하게 '한 칸 이동'과 같은 비용으로 오갈 수 있습니다. 작업자 캐릭터들이 할 일이 하나 더 늘어난 셈이기는 하지만, 적재적소에 건설해 둔 텔레포터는 캐릭터들의 활동 범위, 자원들의 가용 범위를 획기적으로 넓히는 중요한 효과를 발휘하게 될 것입니다.

요청 게시판

거점과 텔레포터는 자원 흐름을 위해 꼭 필요한 시설들이지만, 건설에 많은 자원이 필요하므로, 내 캐릭터 혼자 건설에 임하는 것은 썩 효율적이지 않습니다. 다행히도, 내 캐릭터가 판단하기에 현재 칸에 새로운 시설을 건설할 필요가 있다면 공개 요청 게시판에 새로운 요청을 게시할 수 있습니다(속칭 '도배'를 막기 위해 한 번 요청을 게시하면 그 목표가 달성될 때까지는 다른 요청을 게시할 수 없게 되어 있으니, 주변을 잘 확인하고 꼭 필요하다 싶을 때만 이용해 주세요). 반대로, 광활한 평면 위 어디로 갈 지 감이 안 오는 경우에는 요청 목록을 확인하여 다른 캐릭터를 돕는 것을 주 목표로 삼아도 좋을 것입니다. 이러한 이타적 활동들을 포함하여 각 캐릭터가 수행하는 행동들에 대해 다양한 관점에서 점수를 잴 예정이니, 앞에서 나온 두 가지 스타일 중에 하나를 고른 다음에는 어떤 부문 점수를 높이기 위해 노력할 것인지 결정해 보아도 좋을 것 같습니다.

점수 집계, 이력 기록 시스템

한 게임에서 캐릭터들은 각각 수천 번, 많으면 수만 번 의사 결정을 수행하게 되며, 그 이력을 모두 확인해 가며 내 캐릭터의 성능을 파악하는 것은 쉬운 일이 아닐 것입니다. 이를 감안하여, 내 캐릭터가 점유 행동을 수행한 횟수나 수집한 총 자원량과 같은 직접적인 항목들, 내 캐릭터가 수집한 자원이 어느 용도로 사용되었는지와 같은 간접적인 항목들, 그리고 의사 결정중 오류를 일으킨 횟수와 같은 로그 측면의 항목들을 포함하여 다양한 부문에 대해 점수를 잴 예정입니다. 이런 점수들을 게임 진행 도중 의사 결정 과정에서 직접 활용할 수도 있고(다만, 강화된 개인정보 보호법에 따라 게임 도중에는 다른 캐릭터의 점수를 알 수 없어요), 게임 종료 이후에 다음 버전 캐릭터를 만들기 위한 참고자료로 활용하는 것도 가능합니다(별도의 파일에 출력해 줄 거예요).

제출 전에도 협력 플레이를 해 볼 수 있도록 제공할 예시 캐릭터들

'본 게임'을 진행할 세계는 제출 기한이 지나 모든 캐릭터가 모인 이후에나 완성됩니다. 이를 감안하여, 캐릭터의 협력 능력을 미리 테스트할 수 있도록 예시 캐릭터들을 마련해 두었습니다. 예시 캐릭터들은 단순한 의사 결정을 수행하며 게임에 참여하도록 구성되어 있으며, 절대 오류를 일으키지 않으므로 본격적으로 내 캐릭터를 만들기 위한 복붙 시작점으로 삼을 수 있습니다.

예시 캐릭터들은 며칠 정도 뒤에 공개될 예정입니다. 이번 과제는 기간이 긴 편이니, 일단 첫 주에는 수업 영상과 설명서들을 읽으며 상상하는 시간을 갖고, 내 캐릭터의 스타일이 어느 정도 결정된 이후부터 예시 캐릭터들을 함께 추가해서 게임을 진행해 보는 것을 추천합니다. 출력된 점수 및 이력들을 체크하면서, 작은 능력치 차이, **수식** 차이가 가져오는 나비 효과를 감상해 보는 것도 좋겠습니다. 이렇게 내 캐릭터의 스타일을 다각도로 점검해 보고, 가장 마음에 드는 플레이를 펼친 버전을 골라 제출하도록 노력해 보세요.

마감 이후에, 모든 캐릭터들이 모여 서로 도와 가며 자원을 모으고 사용하는 세계를, 그리고 그 안에서 나름의 '1인분'을 하고 있는 내 캐릭터를 보면, 이번 과제에다 쏟은 노력에 대한 소소한 뿌듯함을 느낄 수 있을 것입니다!

의사 결정과 Data

캐릭터가 다룰 수 있는 Data

의사 결정을 수행하면서 내 캐릭터는 꽤 다채로운 Data를 읽고 사용할 수 있습니다. 예를 들면, 누군가가 게시한 요청을 읽고 각 텔레포터의 위치를 체크하며 그 칸으로 최대한 빨리 이동하기 위해 지금 당장 무슨 행동을 할 지 결정할 수 있습니다. 따라서, 본격적으로 게임 규칙들을 살펴보기 전에 먼저 내 캐릭터가 다룰 수 있는 Data를 구경해 보고 가면 좋을 것 같습니다.

내 캐릭터는 의사 결정 도중 아래와 같은 Data를 제한 없이 사용할 수 있습니다(이것들 말고 몇 가지 더 있기는 해요. 자세한 내용 및 **수식** 적는 방법은 다른 설명서에 적어 두었어요):

- 캐릭터들에 대한 Data
  - 내 캐릭터의 이름과 각종 능력치들, 내 캐릭터가 들고 있는 자원의 양, 각 부문별 점수
  - 모든 캐릭터들의 현재 위치
- 내 개인 평면에 대한 Data(거리와 무관하게 모든 칸에 대한 Data를 항상 읽을 수 있어요)
  - 각 칸에 놓인 자원의 양
- 공유 평면에 대한 Data(역시나, 거리와 무관하게 항상 읽을 수 있어요)
  - 각 칸에 대한 점유 및 건설 현황(완료되었는지, 아니라면 자원이 얼마나 더 필요한지)
  - 각 칸에 캐릭터가 몇 명 있는지
  - 각 칸의 거점에 수납되어 있는 자원의 양
  - 평면 위의 모든 거점 및 텔레포터 위치 목록
- 요청과 관련한 Data
  - 내 캐릭터가 마지막으로 게시한 요청
  - 게시했고 유효한(아직 달성되지 않은) 요청 목록, 이제까지 게시된 모든 요청 목록
  - 어떤 요청이 달성되었는지 여부, 달성하기까지 남은 자원의 양

여기에 더하여, 이번에 **계산**해 둔 Data를 다음 의사 결정에 활용하기 위해 내가 별도로 정한 **이름**에 담아 두는 것도 가능합니다. 이전과는 다르게 이번 과제에서는 여러분이 생각하는 특정 목표를 달성하기 위해 여러 행동들을 **순차**적으로 수행해야 할 가능성이 높습니다(예를 들어, 다른 거점으로 가기 위해 이동 및 워프를 여러 번 수행하거나, 거점 건설을 완료하기 위해 이동 → 자원 획득 → 다시 이동 → 건설 행동을 수 번 내지 수십 번 수행해야 할 수 있습니다). 그렇기에 보통은 '내가 뭐 하려고 했지?'에 대한 Data를 추가로 다루어 가며, 이를 바탕으로 위에 나열되어 있는 각종 Data를 조합하여 '그러면 이번엔 무슨 행동을 해야 하지?'를 결정하게 됩니다. 이러한 점을 염두에 둔 채로, 다음 페이지 내용을 함께 확인해 봅시다.

캐릭터가 할 수 있는 행동들(의사 결정 함수가 return하는 Data가 갖는 의미)

내 캐릭터는 매 의사 결정마다 아래에 나열된 열 가지 행동들 중 하나를 선택할 수 있습니다:

#	명칭	설명
0	수집	개인 평면 위 현재 칸에 놓인 자원을, 가능한 최대치만큼(손이 충분하면 전부 다, 아니면 들 수 있는 만큼) 집어 듭니다.
1	이동	상/하/좌/우 중 원하는 방향으로 한 칸만큼 걸어갑니다. 자원을 든 채 이동할 수 있습니다.
2	점유	들고 있는 자원을 모두 사용하여 현재 칸에 대한 점유에 기여합니다.
3	거점 건설	현재 칸에 거점이 아직 없다면, 들고 있는 자원을 모두 사용하여 거점 건설에 기여합니다.
4	텔레포터 건설	현재 칸에 텔레포터가 아직 없다면, 들고 있는 자원을 모두 사용하여 텔레포터 건설에 기여합니다.
5	수납	현재 칸에 거점이 있다면, 들고 있는 자원을 모두 거점에 내려놓습니다.
6	인출	현재 칸에 거점이 있다면, 거점에 수납된 자원을 가능한 최대치만큼(손이 충분하면 전부 다, 아니면 들 수 있는 만큼)집어 듭니다.
7	워프	현재 칸에 텔레포터가 있다면, 텔레포터가 있는 아무 칸으로나 워프할 수 있습니다.
8	요청 게시	자신이 게시한 유효한 요청이 없는 경우, 현재 칸에 대한 새로운 요청을 게시합니다.
9	대기	현재 칸에서 잠시 쉽니다. '기력'이 회복됩니다. 현재 칸에 거점이 있다면 더 많은 기력을 회복할 수 있습니다.

매 의사 결정을 수행할 때마다, 위의 열 가지 숫자 **값**들 중 하나를 return함으로써 내 캐릭터가 지금부터 어떤 행동을 수행할 것인지를 결정하게 됩니다. 대부분의 행동들은 위에 나열한 순서에 해당하는 숫자 하나만 골라 return하면 그 행동을 선택한 것으로 간주됩니다. 이동, 워프, 그리고 요청 게시 행동의 경우, 어느 방향, 어느 좌표, 어떤 행동에 대한 요청을 할 것인지 추가로 지정하여 return하게 됩니다(Python에서는 값 여러 개를 손쉽게 한 묶음으로 return할 수 있도록 되어 있으니 걱정 안 해도 돼요).

각 행동들은 수행하며 소모하는 비용이 다르며, 세부적인 **값**은 캐릭터별, 칸별, 상황별 Data에 의해 달라질 수 있습니다. 각 행동에 대한 보다 자세한 내용은 다음 설명서에서 확인하기로 하고, 일단 지금은 Data 관련 둘러보기를 마치기 위해 내 캐릭터의 특성을 지정하기 위해 어떤 Data를 직접 정할 수 있는지 한 번 구경해 봅시다.

내 캐릭터를 구성하기 위해 여러분이 정할 수 있는 Data

여러분은 자신만의 **모듈**을 구성하면서 내 캐릭터를 묘사하기 위해 아래 나열된 네 가지에 해당하는 **값**이 담긴 `list`를 하나 지정하게 됩니다(**계산**하면 그 `list`가 나오는 **수식**을 적어 **할당문**을 구성하면 간단히 끝낼 수 있어요):

0. 이름: 여러분의 인성을 마음껏 뽐낼 수 있는 강렬한 `str` **형식 값**입니다
    - 이름은 주로 프로그램을 실행하는 사용자에게 출력하기 위한 용도로 사용합니다. 게임 진행 도중에는 대부분 이름 대신 **index**를 사용합니다

1. 힘(STR) 능력치: 자원을 들고 수행하는 행동의 효율에 영향을 줍니다.
    - 수집가 스타일 캐릭터에게 중요한 능력치입니다
    - 수집, 인출 행동을 통해 들고 있을 수 있는 자원의 최대치를 결정합니다
      (힘이 10이면 최대 10만큼 들 수 있어요)
    - 자원을 들고 이동, 워프할 때의 '기력' 소모량을 줄일 수 있습니다

2. 민첩성(DEX) 능력치: 자원을 들지 않고 수행하는 행동의 효율에 영향을 줍니다.
    - 작업자 스타일 캐릭터에게 중요한 능력치입니다
    - 점유, 거점 건설, 텔레포터 건설 행동의 '후딜레이'를 대폭 줄입니다
    - 자원을 들지 않은 채로 이동, 워프할 때의 후딜레이를 그럭저럭 줄입니다

3. 체력(CON) 능력치: 기력 소모량이 큰 행동들을 연거푸 수행할 때의 효율에 영향을 줍니다.
    - 기존 거점에서 멀리 이동하여 활동하려는 캐릭터에게 중요할 수 있는 능력치입니다
    - 대기를 제외한 모든 행동을 할 때 소모되는 기력의 최대치를 결정합니다
    - 대기 행동의 기력 회복량을 증가시킵니다(최대치 비례 회복이라 그래요)

이번 게임에도 능력치 지정과 관련한 조건이 들어 있습니다. 구체적으로, 캐릭터의 세 능력치 **값**은 `int` **형식**이어야 하며, 그 합이 항상 30이어야 합니다(이번에는 모든 능력치 **값**이 양수여야 한다는 조건까지 추가해 두었어요). 아마도 여러분은 기본적인 10/10/10 조합에서 시작해 자신이 추구할 스타일에 따라 조금씩 조절해 가며 마음에 드는 조합을 찾게 될 것입니다. 일단은 기본 조합으로도 무난하게 과제 점수 만점을 받을 수 있도록 해 둘 예정이니, 먼저 의사 결정용 **함수** 구성을 얼추 끝낸 다음에 세부 성능을 튜닝하는 느낌으로 다양한 능력치 조합을 적용하여 테스트해 보는 것을 추천합니다.

각 능력치에 대한 구체적인 효과, 스타일에 따른 능력치 지정 가이드는 다음 설명서에 적혀 있습니다. 일단 여기서는, 이전 페이지에서 등장했던 '후딜레이'와 '기력'에 잠시 주목해 봅시다.

(중요)Data와 행동 효율#1 – '후딜레이' 이야기

행동의 효과는 그 행동에 대한 의사 결정이 끝난 직후에 적용됩니다. 예를 들어, 내 캐릭터가 이번에 수납 행동을 하기로 결정했다면 들고 있던 자원을 그 즉시 거점에 놓으며, 따라서 바로 뒤에 의사 결정을 수행하는 캐릭터부터 그 자원을 인출할 수 있게 됩니다.

이렇게 행동이 끝난 이후에는 그 다음 의사 결정을 언제 할 것인지가 결정되며, 이 게임에서는 이러한 두 의사 결정들 사이의 시간 간격을 후딜레이라고 부릅니다(비속어기는 한데 직관적이라 그냥 쓰기로 했어요).

어떤 행동들은 후딜레이가 능력치와 무관하게 일정하며(수집 행동의 경우 보통은 1초 걸려요), 몇몇 행동들은 민첩성 능력치에 따라 후딜레이가 큰 폭으로 변할 수 있습니다(점유 행동의 경우 민첩이 1이면 6초, 10이면 0.45초, 19면 0.03초 걸려요). 그렇기는 하나, 세 능력치의 합은 30으로 고정되어 있기 때문에, 작업자 스타일을 추구한다 하더라도 민첩성 능력치만 너무 높게 지정하면 작업 한 번에 사용 가능한 자원의 양이 너무 적거나 중간중간 기력 회복을 위해 대기 행동을 더 자주 수행해야 할 수도 있습니다.

반면 수집가 스타일을 고른 경우에는, 자원을 들지 않은 채로 이동, 워프할 때 민첩성 능력치에 따르는 후딜레이 감소 효과를 볼 수 있으나(민첩이 1이면 0.9초, 10이면 0.35초, 19면 0.14초) 그 감소량이 크지 않고, 수집가가 자주 수행하게 될 수집, 수납 행동은 후딜레이가 능력치의 영향을 받지 않으므로, 이동 후딜레이를 줄이기보다는 힘 및 체력 능력치를 확보하여 자원을 더 많이, 더 자주 들어 옮길 수 있도록 하는 것이 더 효율적일 수 있습니다. 다만, 한 번 수집한 자원을 다시 개인 평면 위에 놓는 것은 불가능하므로, 주변에 거점이 없다면 손에 든 자원을 소모하기 위해 점유 또는 건설 행동을 선택하게 될 수 있습니다. 따라서 민첩성 능력치를 낮게 지정한 수집가 캐릭터들은 거점 하나를 정해 그 주변에서 활동하도록 의사 결정을 수행할 필요가 있을 것입니다.

Data와 행동 효율#2 – '기력' 이야기

대기 행동을 제외한 모든 행동은 기력을 소모합니다. 대부분의 행동은 기력 소모량이 고정되어 있으며, 집어 들거나 사용하는 자원의 양 또한 소모량에 영향을 주지 않습니다. 예외적으로 이동 및 워프 행동의 경우에는 들고 있는 자원의 양과 힘 능력치에 따라 실제 소모량이 결정됩니다만(자원량 - 힘 // 2만큼 소모, 최소 1 소모), 이는 항상 10만큼 소모하는 수집, 점유 행동이나 항상 20만큼 소모하는 거점 건설, 텔레포터 건설 행동에 비하면 적은 수준입니다. 그러므로 수집할 수 있는 칸이 주변에 여럿 존재하는 경우 가급적 더 많은 자원을 집어 들 수 있는 칸(들)을 선택하는 것이 보통은 더 효율적입니다.

줄어든 기력은 대기 행동을 수행하여 회복할 수 있습니다. 최대 기력 및 기력 회복량은 체력 능력치에 정비례하며, 거점이 있는 칸에서 대기하는 경우 회복 속도가 두 배가 됩니다. 대기 행동은 항상 1초의 후딜레이를 가지므로, 거점 주변에서 활동할 때는 언제 기력을 회복할 것인지를 고려하여 동선을 계획하는 것이 전반적인 효율을 높이는 데 도움을 줄 것입니다.

세계를 움직이는 Code

Data와 관련한 설명은 이 정도로 해 두고, 이번에는 게임의 무대인 다중 평면 세계를 구성하는 Code **실행** 흐름을 간단히 구경해 봅시다(이렇게 적긴 했는데 결국 또 반쯤 Data 이야기긴 해요).

기본 흐름

게임 한 판은 아래와 같은 순서로 진행됩니다:

0. 각종 **모듈**들을 import하면서 게임에서 사용할 Data들을 미리 준비해 둡니다.
1. 모든 캐릭터들을 중심 칸에 배치하고 첫 의사 결정 순서를 정합니다.
2. 활동을 시작하기 직전에 마지막 준비를 진행할 수 있도록 각 캐릭터의 Data 초기화용 **함수**를 한 번씩 호출합니다.
3. 게임 종료 조건('본 게임'의 경우 10,000초 경과)을 만족할 때까지...
   - 이번에 행동할 캐릭터의 의사 결정 **함수**를 호출합니다
   - Return된 의사 결정 결과를 토대로 행동을 수행, 결과를 적용합니다
   - 다음에 의사 결정을 수행할 캐릭터를 결정합니다
4. (게임 종료 이후)게임 결과를 확인하기 위한 내용들을 각각의 파일에 기록합니다.

보다시피, Data 부분에 비하면 이 게임의 Code 부분은 이전 과제의 것과 사실상 동일합니다. 캐릭터 하나를 만드는 여러분은, 내가 구성해 둔 **함수**가 적절한 시점에 호출된다는 점, 그 때마다 내 캐릭터가 지금부터 어떤 행동을 수행할 것인지를 결정하여 적절한 **값**을 return하면 된다는 점만 기억하면 됩니다. 뭐 Python 동네에서는 매일같이 하게 될 일이니(후반부 프로젝트 진행할 때도 이 때와 동등한 느낌으로, 주기적으로 호출될 **함수**에 대한 **함수 정의** 내용물을 채워서 전체 GUI 프로그램을 구성하게 될 거에요), 한 번 더 연습해 본다 생각하고 도전해 보면 좋겠습니다.

게임의 mode – 본 게임과 각종 튜토리얼 시나리오들

과제 점수 산정 및 각종 시상의 대상이 되는 본 게임은 제출 마감 이후 여러분의 캐릭터들이 모두 모인 다음에나 진행할 수 있기에, 그 전까지는 아무도 본 게임이 어떤 양상으로 흘러갈 지 알 수 없습니다. 이 점을 감안해서 이번 과제에서는, 여러분이 조금 더 수월하게 게임 규칙들을 이해하고 자신의 캐릭터를 튜닝해 볼 수 있도록 몇 가지 시나리오들을 마련해 두었습니다. 과제 진행 기간 동안 몇 가지 시나리오들, 테스트용 모드들을 더 추가할 예정이니, 이 설명서를 처음 보고 있는 지금 시점에는 일단 수업 영상을 참고해 가며 기본기를 점검하고, 궁금한 점이 있다면 강사에게 문의해 가며 내 캐릭터의 스타일을 어떻게 가져갈 것인지 결정해 봅시다.

여기까지 읽으면 게임 전반에 대한 대략적인 설명을 다 구경한 셈이 돼요.