컴퓨팅 사고력 개발을 위한 WeDo 2.0 프로젝트 활용

이번 장에서는 과학적인 맥락에서 컴퓨팅 사고력 개발을 위해 WeDo 2.0을 활용하는 방법을 살펴볼 것입니다.

컴퓨팅 사고력 개발을 위한 LEGO® Education WeDo 2.0 프로젝트 활용

LEGO® Education의 프로젝트는 학생들의 컴퓨팅 사고력 개발을 돕기 위한 목적으로 초등 과정의 수업에 사용할 수 있도록 특별히 구상되었습니다.

컴퓨팅 사고력은 일상적인 문제의 해결을 위해 누구나 사용할 수 있는 유용한 기술입니다. WeDo 2.0는 모든 프로젝트마다 단계별로 이러한 기술의 개발을 지원하며, 프로젝트별로 중점 개발 영역이 표시되어 있어 교사의 판단에 따라 학생들의 요구 사항에 가장 적합한 프로젝트를 선택할 수 있습니다.

WeDo 2.0의 모든 프로젝트는 레고 블록과 프로그래밍 언어를 조합하여 사용하도록 구상되었으며, 학생들이 프로그래밍의 원리를 배우는 동시에 문제에 대한 해법을 찾을 수 있도록 해 줍니다.

WeDo 2.0은 코딩 활동을 통해 컴퓨팅 사고력의 개발을 지원함으로써 학생들의 창작품에 생명을 불어넣고 얼굴에는 미소를 안겨주며 새로운 발견에 대한 의욕을 자극합니다.

컴퓨터 과학, 컴퓨팅 사고력 및 코딩

인류사 초반에 태동한 과학 및 엔지니어링 분야와는 달리 컴퓨터 과학은 그 역사가 비교적 짧습니다. 그럼에도 컴퓨터 과학은 과학 및 엔지니어링에 대한 접근 방식뿐 아니라 우리의 생활 방식에까지 광범위하게 영향을 미치고 있습니다.

컴퓨터 과학은 STEM 과목 중 하나로서, 과학, 기술, 공학 및 수학의 특성을 모두 공유하고 있습니다.

모든 STEM 과목은 사고 방식과 평생의 실행력을 개발할 수 있는 기회를 제공하며, 이러한 방법론을 기반으로 질문을 하고 해법을 구상하고 결과를 소통할 능력을 찾을 수 있습니다.

컴퓨팅 사고력은 이러한 방법론의 영역을 넓혀주며, 생각의 방식인 동시에 누구나 문제의 해결을 위해 사용하는 방식이기도 합니다.

컴퓨팅 사고력은 여러 기술의 집합체로 묘사될 수 있으며, 그러한 기술 중 하나가 바로 알고리즘적 사고입니다. 그리고 알고리즘을 생성하는 행동을 “코드” 또는 “코딩”이라고 부릅니다.

따라서 코딩은 STEM의 맥락에서 컴퓨팅 사고력을 개발하기 위한 하나의 수단이 됩니다.

STEM 과목

과학, 기술, 공학, 수학, 컴퓨터 과학

사고방식과 평생의 실행력 개발

1. 질문하기와 문제 해결.
2. 모델 활용.
3. 프로토타입 설계.
4. 조사.
5. 데이터 분석 및 해석.
6. 컴퓨팅 사고력 활용.
    a. 문제분해
    b. 추상화
    c. 알고리즘적 사고(코드)
    d. 평가
    e. 일반화
7. 증거에 기초한 논지 수립.
8. 정보 수집, 평가 및 소통.

컴퓨팅 사고력이란?

"컴퓨팅 사고력" 이라는 표현은 Seymour Papert가 처음 사용하였으나, 그 아이디어를 널리 보편화한 인물은 Jeannette Wing 교수로 알려져 있습니다. Wing 교수는 컴퓨팅 사고력을 이렇게 정의했습니다.

"정보처리 에이전트에 의해 효과적으로 실행될 수 있는 형식으로 해결책이 제시될 수 있도록 문제와 그 해법을 표현하는 과정에 수반되는 사고 과정." (Wing, 2011)

컴퓨팅 사고력은 다양한 분야와 상황에서 사용되며, 우리의 일상 생활에도 흔히 적용됩니다. 무엇보다 컴퓨팅 사고력은 과학, 공학 및 수학 분야에서 널리 사용되며, 다음과 같이 정의될 수 있습니다.

문제분해

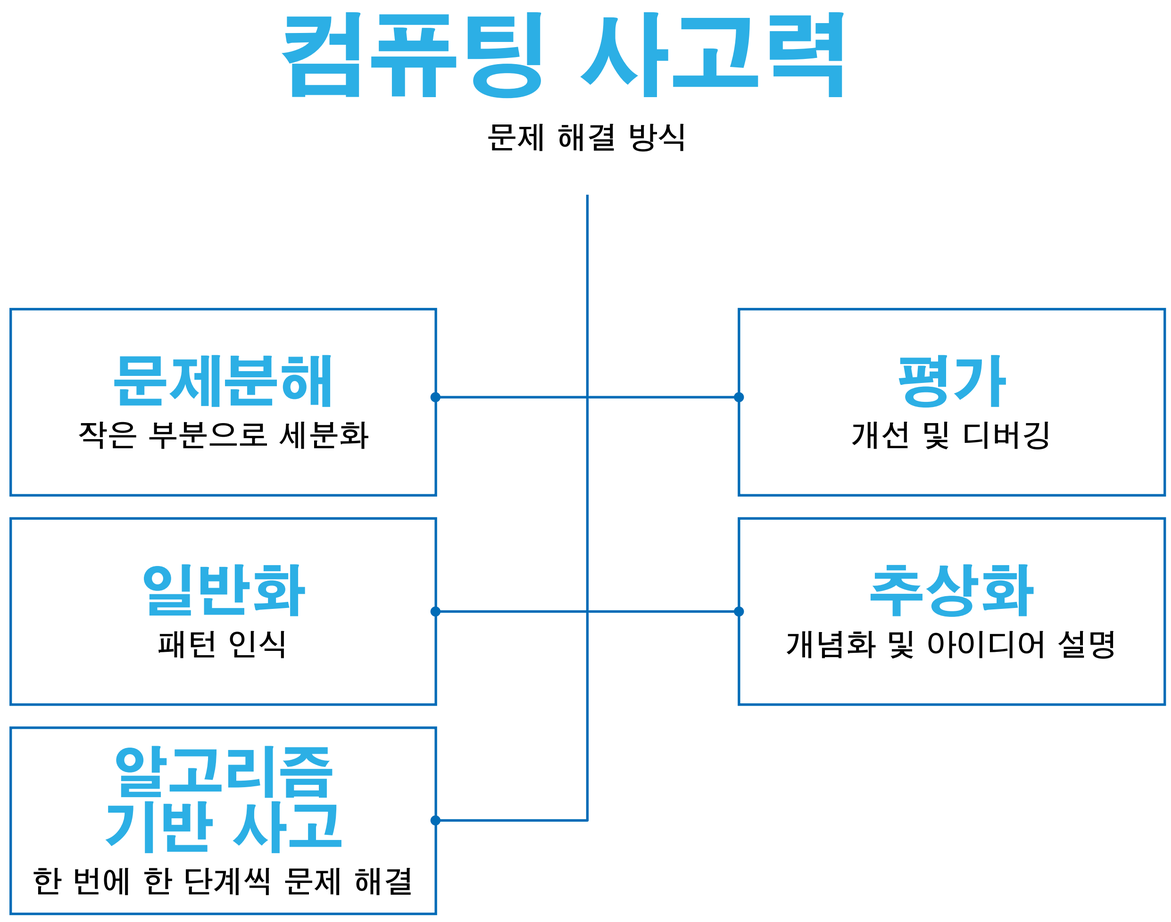
문제분해란 해법을 찾는 과정을 용이하게 하기 위해 문제를 보다 작은 부분으로 나누어 단순화하는 것을 말합니다. 이렇게 하면 문제를 다른 사람에게 설명하거나 개별 과제로 나누기가 한결 쉬워지며, 문제분해의 결과가 일반화로 이어지는 경우가 많습니다.

예: 휴가 여행을 위한 준비 작업(또는 프로젝트)을 항공편 예약, 호텔 예약, 짐가방 꾸리기 등의 하위 작업으로 나눌 수 있습니다.

일반화(패턴 인식)

일반화(패턴인식)란 작업을 분리하여 이미 알려져 있거나 다른 곳에서 보았던 작은 부분으로 나누는 것을 의미합니다. 이렇게 하면 알고리즘을 설계하기가 한결 쉬워집니다.

예: 신호등은 동일한 일련의 동작을 영원히 반복하는 방식으로 작동합니다.



컴퓨팅 사고력이란?

알고리즘적 사고

알고리즘적 사고란 문제의 해결을 목적으로 일련의 순서화된 단계를 만드는 것을 말합니다.

예제 1: 조리법에 따라 요리를 하기 위해 일련의 정해진 단계를 거쳐 음식을 준비합니다.
예제 2: 컴퓨터로 무언가를 하기 위해 컴퓨터가 수행해야 할 일련의 동작을 코딩합니다.

평가 또는 디버깅

프로토타입이 의도대로 작동하는지 여부를 검증하고 뭔가 잘못되었을 경우 어느 부분을 개선해야 할지를 파악하는 능력을 의미합니다. 또한 컴퓨터 프로그래머가 프로그램의 오류를 찾아 수정하기 위해 수행하는 과정을 의미합니다.

예제 1: 요리를 할 때 양념이 바르게 첨가되었는지 여부를 확인하기 위해 수시로 맛을 봅니다.
예제 2: 글을 작성하는 과정에서 철자법 오류 또는 누락된 구두점을 찾기 위해 디버깅을 실시하여 올바르게 읽히도록 만듭니다.

추상화

추상화란 중요하지 않은 세부 요소를 제거하고 문제 또는 해법을 간명하게 설명하는 능력을 말하며, 달리 말해 아이디어를 개념화한다는 것을 의미합니다.

예: 자전거의 일부 요소만으로도 자전거를 묘사할 수 있습니다. 자전거에 관심이 있는 사람이라면 그 유형과 색상, 그리고 몇몇 요소에 대한 설명만 듣고도 전체를 이해할 수 있을 것입니다.

컴퓨팅 사고력 개발을 위한 프로세스

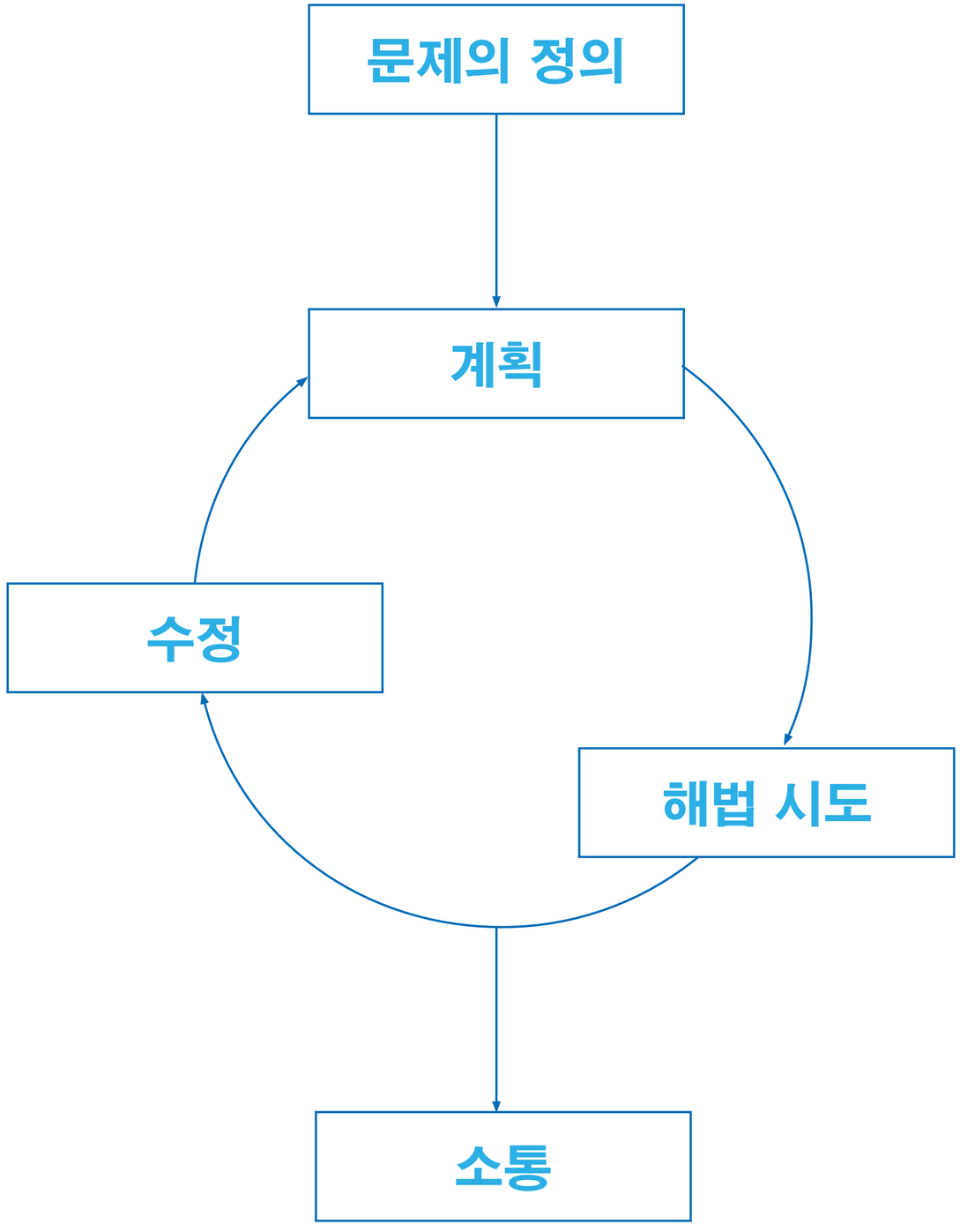
엔지니어링 설계 프로세스의 활용
엔지니어들은 문제를 해결할 방법을 찾기 위해 설계 프로세스를 주로 사용하며, 일련의 단계를 거쳐 해법을 찾아냅니다. 그리고 이러한 단계마다 모종의 기술이 사용되거나 기술이 개발되는데, 이러한 기술을 "컴퓨팅 사고 기술"이라 부릅니다.

WeDo 2.0은 학생들에게 이와 유사한 프로세스를 따를 것을 요구합니다.

문제의 정의
개선을 요하는 문제 또는 상황이 학생들에게 주제로 제시되며, 경우에 따라 많은 양의 세부 정보가 문제에 내포될 수 있습니다. 따라서 해결 방법을 쉽게 하려면 문제를 여러 개의 작은 조각으로 나누어야 합니다.

즉, 문제를 단순한 방식으로 정의하고 성공 기준을 찾아내는 과정을 통해 학생들이 "문제분해"의 기술을 습득할 수 있습니다.

요약 정리:
- *학생들이 스스로의 힘으로 문제를 설명할 수 있는가?*
- *문제해결에 성공했는지 여부를 평가할 수 있는 방법을 학생들이 설명할 수 있는가?*
- *학생들이 문제를 보다 관리하기 쉬운 작은 조각으로 세분화할 수 있는가?*



컴퓨팅 사고력 개발을 위한 프로세스

계획 수립
학생들이 시간을 들여 문제를 해결할 여러 가지 방법을 구상하고, 그중 하나의 아이디어를 실행에 옮기기 위한 상세한 계획을 수립합니다. 아울러 해법에 다다르기 위해 거쳐야 할 제반 단계를 정의하는 과정이 수반되며, 이미 보아 알고 있는 과제들을 식별하는 작업을 통해 이른바 “일반화” 기술을 개발할 수 있습니다.

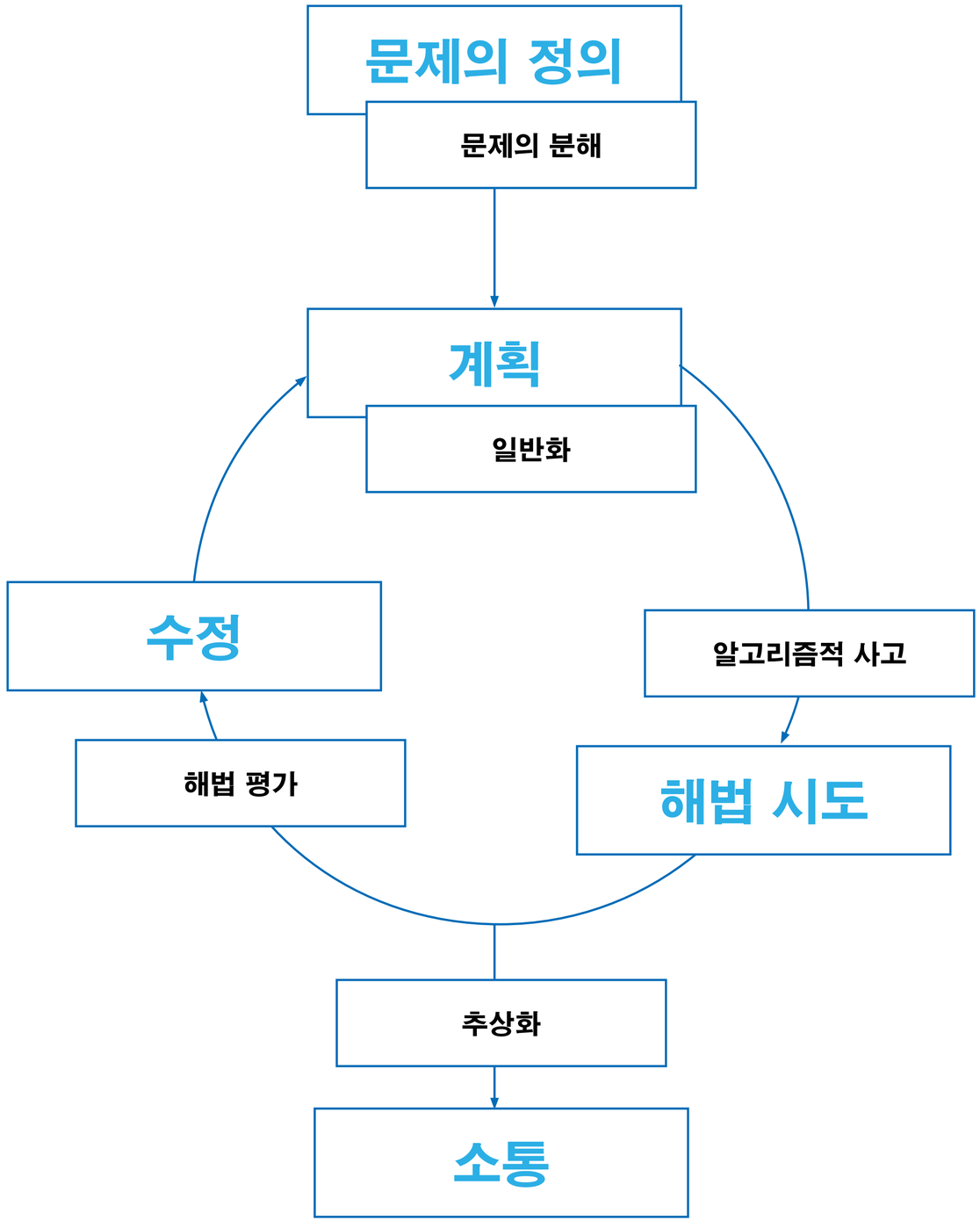
요약 정리:
- *학생들이 프로그래밍할 활동의 목록을 만들 수 있는가?*
- *학생들이 사용 가능한 프로그램의 부분 요소들을 식별할 수 있는가?*
- *학생들이 프로그램의 일부를 재사용할 수 있는가?*

시도
모든 학생에게 해법의 최종 버전을 완성하라는 과제가 부여됩니다. 또한 이 단계에서 학생들이 프로그래밍 언어를 이용해 레고® 모델을 작동하게 되며, 아이디어를 코딩하는 과정에서 알고리즘적 사고 능력이 개발됩니다.

요약 정리:
- *학생들이 해법을 프로그래밍할 수 있는가?*
- *학생들이 순서, 반복 조건문 등을 사용할 수 있는가?*

수정
프로그램과 모델이 성공 기준에 부합하는지 여부에 따라 학생들 스스로 자신이 개발한 해법을 평가합니다. 아울러 학생들이 평가 기술을 사용하여 프로그램의 일부를 변경, 수정, 디버깅 또는 개선해야 하는지 여부를 판정합니다.

요약 정리:
- *학생들이 프로그램 반복 기법을 사용하는가?*
- *학생들이 프로그램의 문제를 수정하는가?*
- *해법이 문제와 연결되어 있는지 여부를 학생들이 판정할 수 있는가?*



컴퓨팅 사고력 개발을 위한 프로세스

소통

학생들이 해법의 최종 버전을 학급 전체에 발표하고 자신의 해법이 어떻게 성공 기준에 부합하는지를 설명하며, 올바른 상세도 수준에서 해법을 설명하는 과정을 통해 추상화 및 소통 기술을 개발할 수 있습니다.

요약 정리:

- 학생들이 *해법의 가장 중요한 부분을 빠뜨리지 않고 설명하는가?*
- *상대방이 알아들을 수 있을 만큼 충분히 자세하게 설명을 하는가?*
- *자신의 해법이 성공 기준에 어떻게 부합하는지를 학생들이 제대로 설명하는가?*

코딩을 통한 컴퓨팅 사고력의 개발

학생들의 알고리즘적 사고 개발을 지원하기 위한 목적으로 몇 가지 프로그래밍 원칙이 소개되며, 해법을 개발하는 과정에서 모델에 생명을 불어넣기 위한 일련의 행동과 구조를 체계화하는 방법을 배울 수 있습니다.

학생들이 가장 공통적으로 사용하게 될 WeDo 2.0 프로그래밍의 원칙은 다음과 같습니다.

1. 출력
출력이란 학생들이 작성하는 프로그램에 의해 제어되는 무언가를 의미합니다. WeDo 2.0의 출력으로는 사운드, 빛, 디스플레이 및 모터 켜기와 끄기 등이 있습니다.

2. 입력
입력이란 컴퓨터 또는 장치가 수신하는 정보를 의미합니다. 이러한 정보는 센서를 사용하여 숫자 또는 텍스트 값의 형태로 입력할 수 있습니다. 예를 들어, 센서가 무언가(거리 등)를 탐지하거나 측정하는 순간 해당 판독 값이 디지털 입력 신호로 바뀌고 프로그램에서 사용할 수 있는 형태로 변환됩니다.

3. 이벤트(대기)
학생들이 프로그램으로 하여금 무언가 기다리던 사건이 발생할 경우 일련의 동작을 계속하도록 명령을 내릴 수 있습니다. 또한 지정된 길이의 시간 동안 기다리거나 센서가 무언가를 감지할 때까지 기다리도록 프로그램을 작성할 수 있습니다.

4. 반복
동작을 영구히 반복하거나 지정된 길이의 시간 동안 반복하도록 프로그램을 작성할 수 있습니다.

5. 함수
함수란 특정한 상황에서 함께 사용 가능한 동작의 집합체를 의미합니다. 예를 들어, 불빛을 깜박이기 위한 용도로 사용 가능한 블록의 집합체를 일컬어 “깜박이 함수”라 부를 수 있습니다.

6. 조건
조건은 특정 상황에서만 실행되는 동작을 프로그래밍하기 위한 목적으로 사용됩니다. 프로그램 내부에 조건을 넣는다는 것은 특정 조건이 충족되지 않을 경우 프로그램의 일부가 실행되지 않는다는 것을 의미합니다. 예를 들어, 기울기 센서를 왼쪽으로 기울이면 모터가 작동하고 오른쪽으로 기울이면 모터가 멈추도록 프로그램을 만들 수 있으며, 이러한 경우 기울기 센서가 영영 왼쪽이나 오른쪽으로 기울지 않는다면 모터가 절대 작동하거나 멈추지 않을 것입니다.