

# 작업 3-2: While 루프 작성

식 복사

## 기본 While 함수 작성

0부터  $n$ 까지의 모든 숫자를 합하는 함수  $\sigma$ 를 작성합니다.

1. 변수  $n$ 이 인수인 함수  $\sigma$ 를 정의하고 새 프로그램을 작성합니다.


$$\sigma(n) := \parallel \parallel$$

2. 빈 **while** 루프를 추가하려면 수학 탭의 연산자 및 기호 그룹에서 **프로그래밍**을 클릭한 다음 **while**을 클릭합니다.

$$\sigma(n) := \parallel \text{while } \parallel \parallel$$

3.  $n > 0$ 인 동안 **while** 루프가 계속 실행되고 루프 내에서 1씩  $n$ 을 감소시키도록 지정합니다.

$$\sigma(n) := \parallel \text{while } n > 0 \parallel \parallel n \leftarrow n - 1 \parallel$$


 **for** 루프와 달리 **while** 반복 변수를 직접 증가시키거나 감소시켜야 합니다.

4. 현재 반복 변수 값을 합계에 더하기 위해 **while** 문 바로 다음에 아래와 같은 줄을 입력합니다.

$$\sigma(n) := \parallel \text{while } n > 0 \parallel \parallel \begin{array}{l} \text{sum} \leftarrow \text{sum} + n \\ n \leftarrow n - 1 \end{array} \parallel$$

5.  $\text{sum}$ 의 값을 반환합니다.

$$\sigma(n) := \parallel \text{while } n > 0 \parallel \parallel \begin{array}{l} \text{sum} \leftarrow \text{sum} + n \\ n \leftarrow n - 1 \end{array} \parallel \text{sum}$$

 마지막으로 반복 변수를 업데이트합니다. 그렇지 않으면 첫 번째 반복 변수 더하기를 삭제해야 합니다.

6. 5에 대해  $\sigma$ 의 값을 계산합니다.

$$\sigma(5) = 15$$

예상대로 프로그램은 다음 합계와 동일합니다.

$$\sum_{i=1}^5 i = 15$$

식 복사

## Continue 문 추가

루프를 계속 실행하지만 특정 반복을 건너뛰려면 *continue* 문을 추가합니다.

0부터  $n$ 까지에서 17로 나뉘지는 숫자를 제외한 모든 숫자를 합하는 함수를 작성합니다.

1. 위 함수를 복사하고 함수 이름을 *sigma\_not17*로 바꿉니다.

```
sigma_not17(n) := || while n > 0
                  || || sum ← sum + n
                  || || n ← n - 1
                  || sum
```

2. *while* 루프 안에서 *while* 문 아래에 새 줄을 추가합니다.

```
sigma_not17(n) := || while n > 0
                  || ||
                  || || sum ← sum + n
                  || || n ← n - 1
                  || sum
```

3. *if* 문을 추가하고 아래에 식을 입력합니다.

```
sigma_not17(n) := || while n > 0
                  || || if mod(n, 17) = 0
                  || || ||
                  || || || sum ← sum + n
                  || || || n ← n - 1
                  || sum
```

4. 무한 루프를 방지하기 위해  $n$ 을 1씩 감소시킵니다.

5. *continue* 문을 추가하려면 수학 탭의 연산자 및 기호 그룹에서 **프로그래밍**을 클릭한 다음 **continue**를 클릭합니다.

```
sigma_not17(n) := || while n > 0
                  || || if mod(n, 17) = 0
                  || || || n ← n - 1
                  || || || continue
                  || || || sum ← sum + n
                  || || || n ← n - 1
                  || sum
```

6. 16과 17에 대해 *sigma\_not17*의 값을 계산합니다.

$$\text{sigma\_not17}(16) = 136$$

$$\sigma_{not17}(17) = 136$$

## Break 문 추가

식 복사

모든 숫자를 합하고 카운터가 20보다 크면 루프를 종료하는 프로그램을 작성합니다.

1. 변수 *sum*을 정의하고 새 프로그램을 작성합니다.


$$sum := \left| \right|$$

2. 빈 *while* 루프를 추가하려면 수학 탭의 연산자 및 기호 그룹에서 프로그래밍을 클릭한 다음 **while**을 클릭합니다.

$$sum := \left| \begin{array}{l} \text{while } ( ) \\ \left| \right| \end{array} \right|$$

3. *while* 루프가 무한 실행되도록 지정합니다.

$$sum := \left| \begin{array}{l} \text{while } (1) \\ \left| \right| \end{array} \right|$$

 *while* 루프는 괄호 안의 식 값이 0이 아닌 한 계속 실행됩니다.

4. *sum* 및 *i*를 초기화합니다.


$$sum := \left| \begin{array}{l} sum \leftarrow 0 \\ i \leftarrow 0 \\ \text{while } (1) \\ \left| \right| \end{array} \right|$$

5. 루프 안에서 반복 변수 *i*의 값을 변수 *sum*에 지정하고 *i*를 1씩 증가시킵니다.

$$sum := \left| \begin{array}{l} sum \leftarrow 0 \\ i \leftarrow 0 \\ \text{while } (1) \\ \left| \begin{array}{l} sum \leftarrow sum + i \\ i \leftarrow i + 1 \end{array} \right| \end{array} \right|$$

6. *sum*의 값을 반환합니다.

$$sum := \left| \begin{array}{l} sum \leftarrow 0 \\ i \leftarrow 0 \\ \text{while } (1) \\ \left| \begin{array}{l} sum \leftarrow sum + i \\ i \leftarrow i + 1 \end{array} \right| \\ sum \end{array} \right|$$

 현재 이 루프는 무한 루프입니다.

7. 루프를 중단하기 위해  $if\ i > 20$ 을 입력하고 *break* 문을 추가합니다. *break* 문을 추가하려면 수학 탭의 연산자 및 기호 그룹에서 프로그래밍을 클릭한 다음 **break**를 클릭합니다.

식 복사


```
sum :=
|
| sum ← 0
| i ← 0
| while (1)
| | if (i > 20)
| | | break
| | sum ← sum + i
| | i ← i + 1
| sum
```

8. *sum*을 계산합니다.

*sum* = 210


9. 루프를 중단하고 프로그램을 종료하려면 *break* 문을 선택하고 수학 탭의 연산자 및 기호 그룹에서 프로그래밍을 클릭한 다음 **return**을 클릭하여 *break* 문을 *return* 문으로 수정합니다. 아래와 같이 자리 표시자에 *sum*을 입력합니다.

```
sum :=
|
| sum ← 0
| i ← 0
| while (1)
| | if (i > 20)
| | | return sum
| | sum ← sum + i
| | i ← i + 1
| sum
```

 프로그램을 즉시 종료하려면 *return*을 사용합니다.

## 실습

다음 작업으로 이동하기 전에 *while* 루프를 사용하여 계승 함수를 구현하는 함수 *fact(n)*를 작성합니다. *n*이 1보다 큰 동안 계속 실행되는 루프를 정의합니다. 루프 안에서 *n*에 변수 *product*을 곱하고 (계승 결과 저장) *n*을 1씩 감소시킵니다.

 PTC Mathcad에서 프로그램 변수는 기본적으로 0으로 설정됩니다. 프로그램의 시작 부분에서 *product*에 1을 지정해야 합니다. 그렇지 않으면 프로그램이 모든 인수에 대해 0을 반환합니다.

작업 3-3으로 이동합니다.