Chapter 11
함수 활용하기

목차

학습목표

- 지역변수와 전역변수에 대해서 알아봅니다.
- 매개변수에 대해서 이해합니다.
- 함수에서 데이터를 반환하는 방법을 알아봅니다.
- 중첩 함수와 재귀 함수에 대해서 알아봅니다.

Section 01

이 장에서 만들 프로그램

이 장에서 만들 프로그램

1. 단위 환산 프로그램

▪mm 단위의 길이를 입력하면 cm, m, inch, ft로 환산된 길이를 출력하는 프로그램

```
길이(mm)를 입력하세요. 1
1 mm -->  0.1 cm
1 mm -->  0.001 m
1 mm -->  0.03937 inch
1 mm -->  0.003281 ft
```

이 장에서 만들 프로그램

1. 할인된 상품 가격표 출력 프로그램
   - ‘오늘의 할인율’을 입력하면 상품별로 할인된 가격을 출력하는 프로그램
   - 원래 정가와 할인 후 가격을 한번에 볼 수 있음

```
-----------------------------------------
-- 한빛마트 오늘의 할인 가격표 출력 시스템 --
-----------------------------------------
오늘의 할인율을 입력하세요. 15
쌀    : 9900 원 15 %DC -> 8415 원
상추  : 1900 원 15 %DC -> 1615 원
고추  : 2900 원 15 %DC -> 2465 원
마늘  : 8900 원 15 %DC -> 7565 원
통닭  : 5600 원 15 %DC -> 4760 원
햄    : 6900 원 15 %DC -> 5865 원
치즈  : 3900 원 15 %DC -> 3315 원
-----------------------------------------
```

Section 02
지역변수와 전역변수란?

지역변수와 전역변수의 개념

- 변수는 함수 내부에 선언하는 경우와 외부에 선언하는 경우에 따라 지역변수와 전역변수로 구분됨
- 지역변수
  - 함수 내부에서 선언되며, 함수 내부에서만 사용 가능함
- 전역변수
  - 함수 외부에서 선언되며, 함수 내·외부에서 모두 사용 가능함

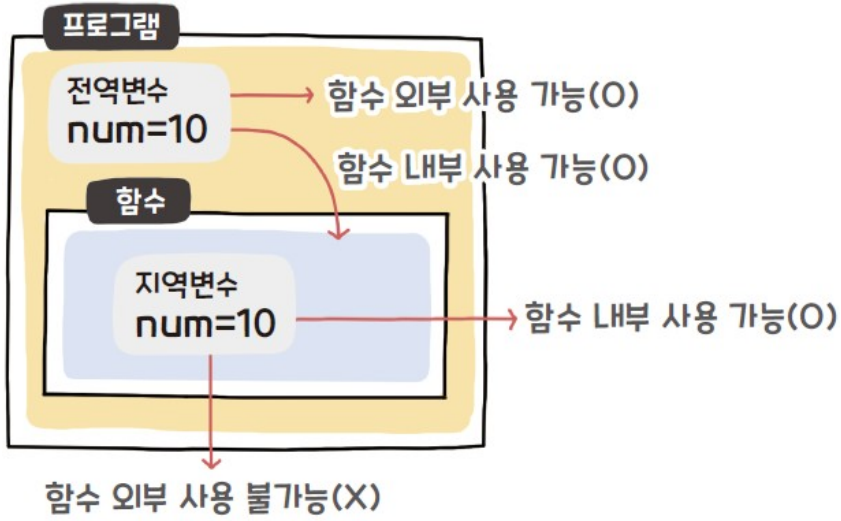


그림 11-1 지역변수와 전역변수의 사용 가능 범위

지역변수와 전역변수의 개념

지역변수와 전역변수의 차이

- 1행에서 선언된 변수 num : 전역변수
- 4행, 6행에서 출력하는 num : 1행의 num과 같은 변수로 전역변수

코드 11-1 ch11_01.py

```
01  num = 10                    # 전역변수 num 선언
02
03  def fun1():                 # 함수 정의
04      print('num : ', num)    # 전역변수 num 사용
05
06  print('num : ', num)        # 전역변수 num 사용
07  fun1()                      # 함수 호출
```

```
num :  10
num :  10
```

전역 변수 num 출력

지역변수와 전역변수의 개념

지역변수와 전역변수의 차이

- fun1() 함수 내부에서 num을 다시 선언하는 경우
- 4행에서 선언된 num : 지역변수
- 1행의 num과 4행의 num은 이름만 같을 뿐 전혀 다른 변수임

코드 11-2 ch11_02.py

```
01  num = 10                    # 전역변수 num 선언
02
03  def fun1():                 ── 함수 정의
04      num = 20                # 지역변수 num 선언
05      print('num : ', num)    # 지역변수 num 사용(함수 안에서 num을 먼저 찾는다.)
06
07  print('num : ', num)        # 전역변수 num 사용
08  fun1()                      ── 함수 호출
```

```
num :  10   ── 전역 변수 num 출력
num :  20   ── 지역 변수 num 출력
```

global 키워드

global 키워드

- '전역을 가리킨다'는 의미로 global 키워드를 사용하여 전역변수에 접근함

→ 'global num'이라고 하면 전역변수 num을 가리킴

코드 11-3 ch11_03.py

```
01  num = 10                          # 전역변수 num 선언
02
03  def fun1():                       # 함수 정의
04      global num                    # 전역변수 num 설정
05      num = 20                      # 전역변수 num 변경
06      print('num : ', num)          # 전역변수 num 사용
07
08  print('num : ', num)              # 전역변수 num 사용
09  fun1()                            # 함수 호출
10  print('num : ', num)              # 전역변수 num 사용
```

```
num :  10
num :  20
num :  20
```

전역 변수 num 출력

문제 해결

문제 해결 11-1 웹사이트의 누적 방문 횟수 프로그램 ch11_sol_01.py

웹사이트 방문 여부를 입력 받아 웹사이트의 누적 방문 횟수를 출력해봅시다.

```
01  flag = True
02  totalVisitor = 0;
03
04  def countVisitor():
05      global totalVisitor
06      totalVisitor += 1
07
08  while flag:
09      inputData = int(input('1.웹사이트 방문, 2.종료 '))
10
11      if inputData == 1:
12          countVisitor()
13          print('누적 방문 횟수 : ', totalVisitor)
14      else:
15          flag = False
```

함수 정의

함수 호출

```
1.웹사이트 방문, 2.종료 1
누적 방문 횟수 :  1
1.웹사이트 방문, 2.종료 1
누적 방문 횟수 :  2
1.웹사이트 방문, 2.종료 1
누적 방문 횟수 :  3
1.웹사이트 방문, 2.종료 2
```

확인문제

확인문제

1. 지역변수와 전역변수에 대한 설명으로 틀린 것은 무엇인가?

   ① 함수 내부에서 선언된 변수를 지역변수라고 한다.

   ② 함수 외부에서 선언된 변수를 전역변수라고 한다.

   ③ 지역변수는 함수 내부에서만 사용할 수 있다.

   ④ 전역변수는 함수 외부에서만 사용할 수 있다.

2. 다음 프로그램의 실행 결과로 옳은 것은 무엇인가?

```
01  totalPrice = 0;
02  goodsCount = [5, 2, 10, 3]
03
04  def printReceipt():
05      goods0Price = 1000
06      goods1Price = 500
07      goods2Price = 2000
08      goods3Price = 1500
09
10      totalPrice += goods0Price * goodsCount[0]
11      totalPrice += goods1Price * goodsCount[1]
12      totalPrice += goods2Price * goodsCount[2]
13      totalPrice += goods3Price * goodsCount[3]
14
15      print('--------- printReceipt ---------')
16      print('totalPrice = ', totalPrice)
17
18  printReceipt()
```

확인문제

①
```
--------- printReceipt ---------
totalPrice =  30500
```

② 에러발생

③
```
--------- printReceipt ---------
totalPrice =
```
에러발생

④
```
--------- printReceipt ---------
totalPrice =  0
```

정답

1. ④

2. ②(`printReceipt()` 함수에서 `totalPrice` 변수를 사용하려면 10행 앞에 `global totalPrice`를 추가해야 함)

Section 03
매개변수와 인수

매개변수의 개념

- 함수 정의부와 함수 호출부
  - 함수 정의부 : 함수를 정의하는 쪽
  - 함수 호출부 : 함수를 사용하려고 호출하는 쪽

- 인수와 매개변수
  - 인수 : 함수를 호출할 때 넘겨주는 데이터
  - 매개변수 : 함수 호출부에서 넘긴 인수를 함수 정의부에서 받아 저장하는 변수

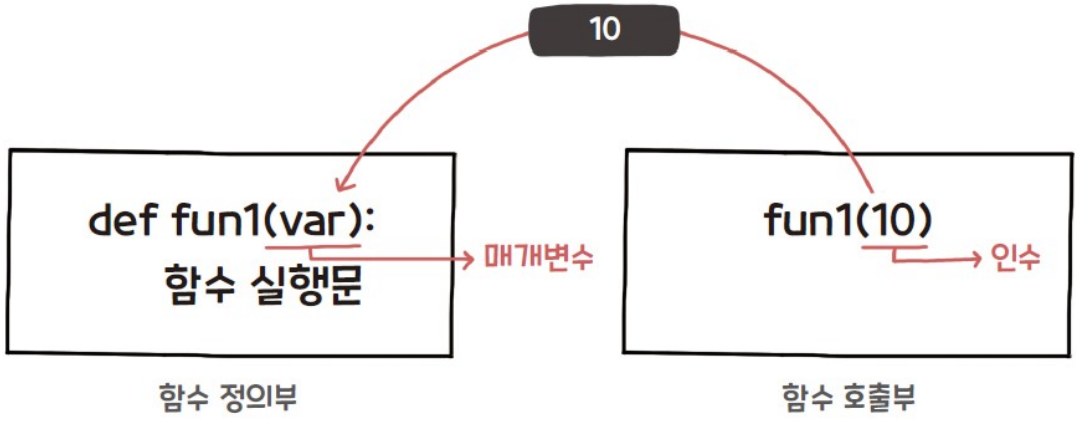


그림 11-2 인수와 매개변수

매개변수의 개념

인수와 매개변수의 사용

코드 11-4 ch11_04.py

```
01  def greet(name):
02      print(name, '씨, 안녕하세요.')
03
04  greet('홍길동')
05  greet('박찬호')
06  greet('박지성')
```

❷ 매개변수에 저장
❸ 출력
❶ 인수 전달

```
홍길동 씨, 안녕하세요.
박찬호 씨, 안녕하세요.
박지성 씨, 안녕하세요.
```

두 개 이상의 매개변수

- 함수 호출부에서 전달할 인수가 두 개 이상인 경우
  - 매개변수는 호출부에서 전달하는 인수의 개수와 순서에 맞춰서 선언
  - 인수와 매개변수가 각각 3개인 코드

코드 11-5 ch11_05.py

```
01  def forecastWeather(temp, humi, rain):      # 인수 3개를 매개변수 3개에 저장
02      print('날씨 예보입니다.')
03      print('최고 온도 : ', temp, '도')
04      print('평균 습도 : ', humi, '%')
05      print('비올 확율 : ', rain, '%')
06
07  temperature = 32
08  humidity = 67
09  rainPercent = 50
10
11  forecastWeather(temperature, humidity, rainPercent)  # 함수 호출 시 인수 3개 전달
```

함수 정의

```
날씨 예보입니다.
최고 온도 :  32 도
평균 습도 :  67 %
비올 확율 :  50 %
```

두 개 이상의 매개변수

하나 더 알기 **인수의 개수와 매개변수의 개수가 일치하지 않는 경우**

인수의 개수와 매개변수의 개수가 일치하지 않으면 에러가 발생합니다. 따라서 전달하는 인수의 개수와 매개변수의 개수는 반드시 일치해야 합니다.

■ 전달하는 인수의 개수가 매개변수 개수보다 적은 경우 → 에러 발생

```
def fun1(n1, n2, n3):          ← 매개변수의 개수 3개
    print(n1, n2, n3)

fun1(10, 20)                   ← 인수의 개수 2개
```

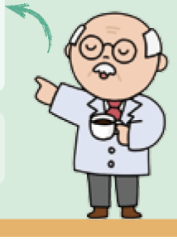
```
TypeError: fun1() missing 1 required positional argument: 'n3'
```

■ 전달하는 인수의 개수가 매개변수 개수보다 많은 경우 → 에러 발생

```
def fun1(n1, n2, n3):          ← 매개변수의 개수 3개
    print(n1, n2, n3)

fun1(10, 20, 30, 40)           ← 인수의 개수 4개
```

```
TypeError: fun1() takes 3 positional arguments but 4 were given
```



인수의 개수를 모르는 경우

- 함수를 호출할 때 전달하는 인수의 개수가 수시로 변경되는 경우
  - 매개변수의 개수도 같이 변경되어야 함
  - '*' 기호를 이용 : 매개변수의 개수를 변경할 필요 없이 해결할 수 있음

- 시험 성적을 출력하는 프로그램
  - 과목수가 3과목에서 4과목으로, 그리고 다시 5과목으로 변경되어도 정상적으로 실행되도록 코딩

인수의 개수를 모르는 경우

코드 11-6 ch11_06.py

```
01  def printAverageScore(*scores):
02      print(type(scores))
03
04      totalScore = 0
05      cnt = len(scores);
06
07      for score in scores:
08          totalScore += score
09
10      print('총점: ', totalScore, '점')
11      print('평점: ', totalScore / cnt, '점')
12      print('-----------------------------')
13
14  printAverageScore(80, 90, 70)
15  printAverageScore(90, 85, 90, 100)
16  printAverageScore(95, 80, 100, 95, 85)
```

함수 정의

```
<class 'tuple'>
총점:  240 점
평점:  80.0 점
-----------------------------
<class 'tuple'>
총점:  365 점
평점:  91.25 점
-----------------------------
<class 'tuple'>
총점:  455 점
평점:  91.0 점
-----------------------------
```

문제 해결

문제 해결 11-2 SMS와 MMS 구별하기 ch11_sol_02.py

문자를 보낼 때 100자 이하인 경우에는 단문 메시지(SMS)로 50원을 부과합니다. 그런데 100자를 넘어가면 장문 메시지(MMS)로 변경되면서 100원이 부과됩니다. 단문과 장문을 구별해서 요금을 부과하는 프로그램을 만들어봅시다.

```
01  def printStringLength(str):
02      strLen = len(str)
03      print('입력한 문자열 길이 : ', strLen)
04
05      if(strLen <= 100):
06          print('단문 메시지로 50원이 부과됩니다.')
07      else:
08          print('장문 메시지로 100원이 부과됩니다.')
09
10  inputData = input('문자열을 입력하세요. ')
11  printStringLength(inputData)
```

함수 정의

```
문자열을 입력하세요. 안녕하세요.
입력한 문자열 길이 :  6
단문 메시지로 50원이 부과됩니다.
```

```
문자열을 입력하세요. 안녕하세요. 저는 대학교 1학년 홍길동입니다...생략...감사합니다.
입력한 문자열 길이 :  344
장문 메시지로 100원이 부과됩니다.
```

확인문제

확인문제

1. 도형의 밑변과 높이를 넣으면 삼각형과 사각형의 넓이를 출력하는 함수이다. 빈칸을 채우시오.

```
01  def printArea(        ,        ):
02      print('삼각형의 넓이 : ', b * h / 2)
03      print('사각형의 넓이 : ', b * h)
04
05  base = int(input('밑변 : '))
06  height = int(input('높이 : '))
07  printArea(        ,        )
```

2. 국어, 영어, 수학 점수를 입력하면 총점과 평균을 출력하는 함수를 포함하는 프로그램이다. 틀린 부분을 찾아 고치시오.

```
01  def printScore(k, e, m):
02      print('총점 : ', k + e + m)
03      print('평균 : ', (k + e + m) / 3)
04
05  korScore = int(input('국어점수를 입력하세요. '))
06  engScore = int(input('영어점수를 입력하세요. '))
07  matScore = int(input('수학점수를 입력하세요. '))
08
09  printScore(k, e, m)
```

정답

1. b, h, base, height 2. 9행을 printScore(korScore, engScore, matScore)으로 수정

인수와 매개변수의 순서가 일치하지 않을 경우

- 호출부에서 전달되는 데이터 순서가 매개변수의 순서와 일치하지 않을 경우
  - 함수 호출부에서 매개변수 이름을 키워드로 명시함

- 회원 정보를 출력하는 프로그램
  - 매개변수 이름을 키워드로 명시함

인수와 매개변수의 순서가 일치하지 않을 경우

코드 11-7 ch11_07.py

```
01  def printMemberInfo(name, email, major, grade):
02      print('name \t:', name)
03      print('email \t:', email)
04      print('major \t:', major)
05      print('grade \t:', grade)
06      print('------------------------------')
07
08  # 매개변수 키워드 사용하지 않음
09  printMemberInfo('Mr. kim', 'kim@gmail.com', 'art', 1)
10
11  # 매개변수 키워드 사용함
12  printMemberInfo(grade=4, major='computer', email='abc@gmail.com', name='Mr. hong')
13  printMemberInfo(major='sport', grade=2, name='Mr. cong', email='def@gmail.com')
14  printMemberInfo(email='ghi@gmail.com', major='music', name='Mr. wang', grade=1)
```

함수 정의

```
name : Mr. kim
email : kim@gmail.com
major : art
grade : 1
------------------------------
name : Mr. hong
email : abc@gmail.com
major : computer
grade : 4
------------------------------
name : Mr. cong
email : def@gmail.com
major : sport
grade : 2
------------------------------
name : Mr. wang
email : ghi@gmail.com
major : music
grade : 1
------------------------------
```

매개변수의 기본값 설정

- 함수 호출에서 매개변수에 전달되는 인수가 없는 경우
  - 매개변수에 기본값을 설정할 수 있음

- 급여 프로그램
  - 관리자의 실수로 급여가 입력되지 않아도 최소 200만원이 지급됨

코드 11-8 ch11_08.py

```
01  def setSalary(name, pay = 200):    # pay는 200으로 기본값이 설정됐다.
02      print(name, '직원 급여 : ', pay, '만원')
03      print('--------------------------------')
04
05  setSalary('박찬호', 400)
06  setSalary('박지성')
07  setSalary('박세리')
```

함수 정의

```
박찬호 직원 급여 :  400 만원
--------------------------------
박지성 직원 급여 :  200 만원
--------------------------------
박세리 직원 급여 :  200 만원
--------------------------------
```

기본값 할당 및 출력

기본값 할당 및 출력

Section 04
데이터 반환

데이터 반환

- return 키워드
  - 함수는 실행이 끝난 후에 나온 결과물(데이터)을 호출부로 반환할 수 있음
  - 이때 사용하는 키워드가 return임

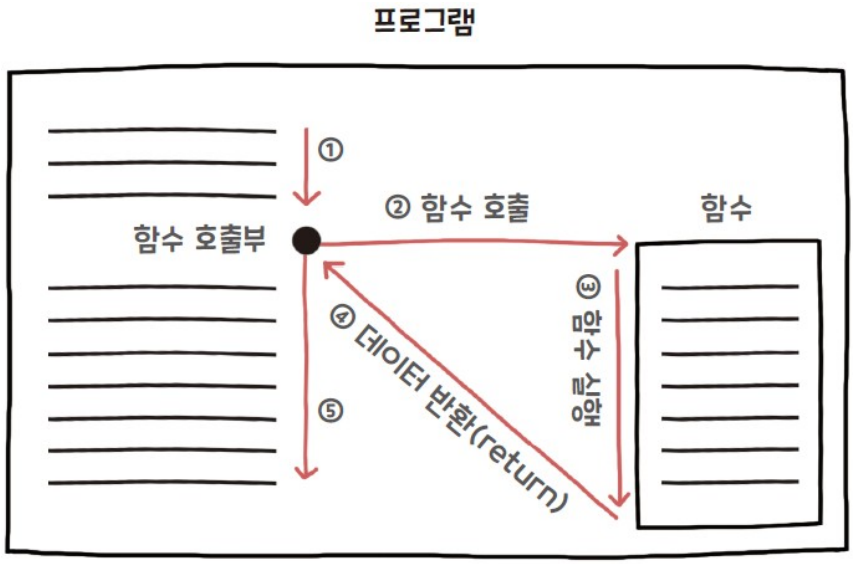


그림 11-3 함수가 포함된 프로그램의 실행 순서

데이터 반환

- return 키워드
  - 덧셈 함수에서 연산한 결과를 호출부로 반환해서 출력하는 프로그램

코드 11-9 ch11_09.py

```
01  def addFunction(n1, n2):
02      sum = n1 + n2
03      return sum
04
05  result = addFunction(10, 20)
06  print(result)
```

❷ 함수 실행
❸ 데이터 반환
❶ 함수 호출

```
30
```

데이터 반환

- return 키워드
  - 함수에서 반드시 사용해야 하는 것은 아님
  - 함수에서 호출부로 돌려줘야하는 데이터가 있을 경우에만 사용
  - [코드 11-9]에서 return을 사용하지 않고 결과를 출력하기

코드 11-10 ch11_10.py

```
01  def addFunction(n1, n2):
02      print(n1 + n2)
03
04  addFunction(10, 20)
```

함수 정의

```
30
```

데이터 반환

- return의 또 다른 기능
  - 함수를 실행 종료함
  - 함수 실행 중 return을 만나게 되면 함수 실행을 중단하고 프로그램은 호출부로 돌아감

코드 11-11 ch11_11.py

```
01  def increaseStar():
02      print('*')
03      print('**')
04      print('***')
05      return
06      print('****')
07      print('*****')
08      print('******')
09      print('*******')
10
11  increaseStar()
```

함수 정의

```
*
**
***
```

문제 해결

문제 해결 11-3 사칙연산 프로그램 만들기 ch11_sol_03.py

2개의 정수를 입력하면 사칙연산(덧셈, 뺄셈, 곱셈, 나눗셈)을 하는 함수를 만들고 그 결과를 튜플로 반환하는 프로그램을 만들어봅시다.

```
01  def calculator(num1, num2):
02      result1 = num1 + num2
03      result2 = num1 - num2
04      result3 = num1 * num2
05      result4 = num1 / num2
06
07      return (result1, result2, result3, result4)
08
09  inputNumber1 = int(input('정수를 입력하세요. '))
10  inputNumber2 = int(input('정수를 입력하세요. '))
11
12  result = calculator(inputNumber1, inputNumber2)
13  print('사칙연산 결과 : ', result)
```

함수 정의

```
정수를 입력하세요. 80
정수를 입력하세요. 60
사칙연산 결과 :  (140, 20, 4800, 1.3333)
```

Section 05
함수 안의 함수

중첩 함수

- 중첩(Nested) 함수
  - 함수 안에 정의된 또 다른 함수

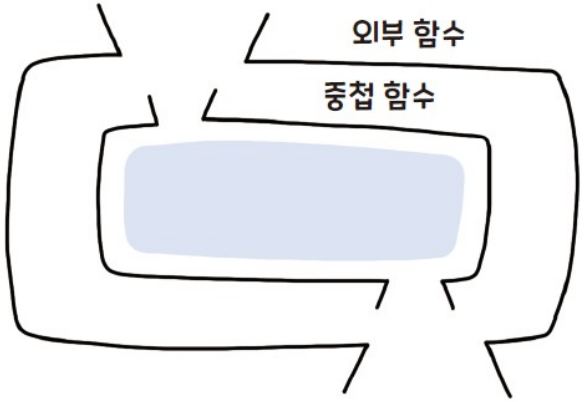


그림 11-4 중첩 함수의 개념

중첩 함수

- 중첩 함수를 이용한 나눗셈 연산 프로그램입니다. divOperator()는

코드 11-12 ch11_12.py

```
01  def divFunction(n1, n2):
02
03      def divOperator(num1, num2):
04          return num1 / num2
05
06      if n2 != 0:
07          result = divOperator(n1, n2)
08      elif n2 == 0:
09          result = '0으로 나눌 수 없습니다.'
10
11      return result
12
13  print(divFunction(10, 0))
14  print(divFunction(10, 2))
```

중첩 함수

❷ 결과 반환

❶ 중첩 함수 호출

외부 함수 정의

```
0으로 나눌 수 없습니다.
5.0
```

중첩 함수

중첩 함수에서 주의할 점

- 내부에 정의되어 있는 중첩 함수는 자신을 포함하고 있는 함수 내부에서만 호출할 수 있으며, 외부에서는 호출할 수 없음

코드 11-13 ch11_13.py

```
01  def divFunction(n1, n2):
02
03      def divOperator(num1, num2):
04          return num1 / num2
05      ... 생략 [코드 11-12]의 6~9행 ...
06
07  print(divFunction(10, 0))
08  print(divFunction(10, 2))
09  print(divOperator(10, 2))
```

중첩 함수를 외부에서 호출하여 에러 발생

```
0으로 나눌 수 없습니다.
5.0
Traceback (most recent call last):
  File "E:\task\writing\pythonEx\temp.py", line 15, in <module>
    print(divOperator(10, 2))
NameError: name 'divOperator' is not defined
```

재귀 함수

- 재귀(Recursive) 함수
  - 함수 안에서 자신을 다시 호출하는 함수
  - 재귀 함수는 자신 안에서 자기 자신을 계속 호출하기 때문에 함수 호출을 종료할 수 있는 코드가 꼭 필요함

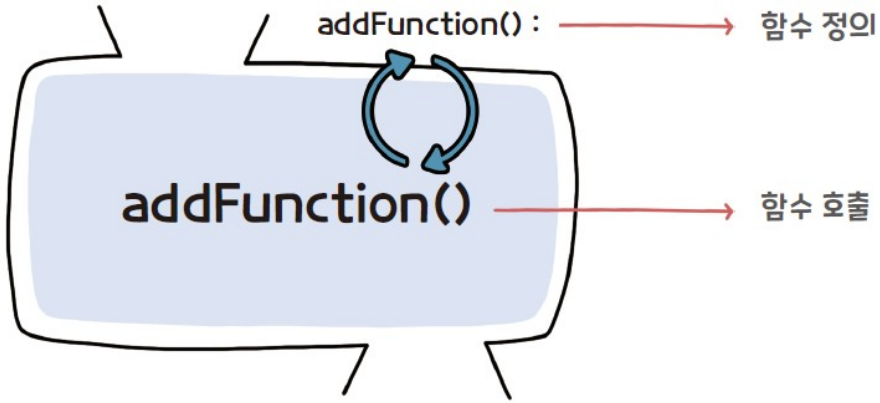


그림 11-5 재귀 함수의 개념

재귀 함수

- 0보다 클 때까지 자기 자신을 계속해서 호출해 1씩 감소하는 프로그램

코드 11-14 ch11_14.py

```
01  # 재귀 함수 정의
02  def addFunction(num):
03      if num > 0:
04          print('num : ', num)
05          addFunction(num - 1)
06
07  addFunction(10)
```

자기 자신을 호출

```
num :  10
num :  9
...생략...
num :  2
num :  1
```

문제 해결

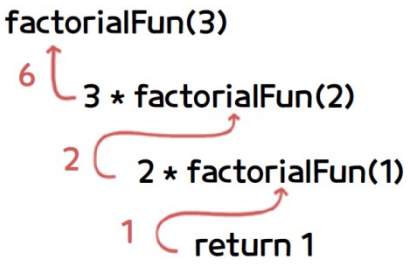
문제 해결 11-4 재귀 함수로 팩토리얼 구현하기 ch11_sol_04.py

사용자가 입력한 정수를 이용하여 팩토리얼 계산을 실행하는 프로그램을 재귀 함수를 이용하여 만들어봅시다.

```
01  # 재귀 함수 정의
02  def fatorialFun(num):
03      if num == 1:
04          return 1
05      else:
06          return num * fatorialFun(num - 1)  ← 재귀 함수 호출
07
08  inputData = int(input('0보다 큰 숫자를 입력하세요. '))
09  result = fatorialFun(inputData)
10  print( inputData, '팩토리얼은 ', result,'입니다.')
```

```
0보다 큰 숫자를 입력하세요. 3
3 팩토리얼은  6 입니다.
```

```
0보다 큰 숫자를 입력하세요. 7
7 팩토리얼은  5040 입니다.
```



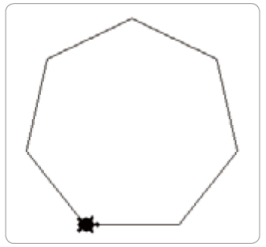
Section 06
터틀 프로그래밍

사용자가 원하는 n각형 그리기

- n각형을 그리기 위해 터틀이 회전해야 할 각도 구함
  - 180-(180*(n-2))/n
- 이후로는 도형을 그릴 때마다 더 이상 복잡한 계산을 할 필요 없이 draw-Shape(n) 함수를 활용함

코드 11-15 ch11_15.py

```
01  import turtle
02  t = turtle.Turtle()
03  t.shape('turtle')
04
05  # 매개변수 값에 따라 도형을 그리는 함수
06  def drawShape(n):
07      angle = 180 - (180 * (n - 2)) / n          # 터틀 회전각도 계산
08
09      for i in range(0, n):
10          t.forward(100)
11          t.left(angle)
12
13  num = int(input('원하는 도형을 입력하세요. '))
14  drawShape(num)
```



```
원하는 도형을 입력하세요. 7
```

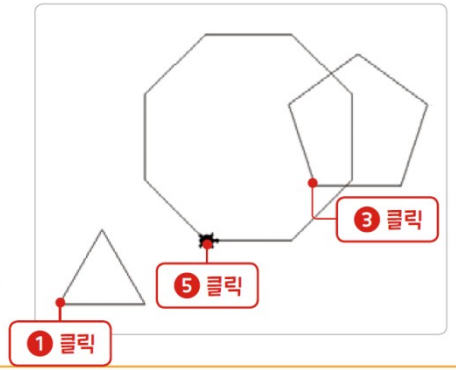
터틀 프로그래밍

- 사용자가 클릭한 곳에 n각형 그리기
  - drawShape() 함수를 업그레이드하여 사용자가 클릭한 곳에 도형 그리기
  - 클릭한 지점의 x, y 좌표를 도화지가 인식할 수 있도록 해야 함
    - onscreenclick() 이벤트를 사용
    - 이벤트의 매개변수로 drawShape를 넣어주면 이벤트가 발생했을 때 draw-Shape() 함수가 호출되면서 x, y 좌표가 도화지에 자동으로 전달됨

터틀 프로그래밍

코드 11-16 ch11_16.py

```
01  import turtle
02  t = turtle.Turtle()
03  t.shape('turtle')
04
05  # 매개변수 값에 따라 도형을 그리는 함수
06  def drawShape(x, y):
07
08      t.up()
09      t.goto(x, y)
10      t.down()
11
12      num = int(input('원하는 도형을 입력하세요. '))
13      angle = 180 - (180 * (num - 2)) / num      # 터틀 회전각도 계산
14
15      for i in range(0, num):
16          t.forward(100)
17          t.left(angle)
18
19  s = turtle.Screen()
20
21  print("도화지에서 원하는 좌표를 클릭하세요.")
22  s.onscreenclick(drawShape)
```




```
도화지에서 원하는 좌표를 클릭하세요.
원하는 도형을 입력하세요. 3
원하는 도형을 입력하세요. 5
원하는 도형을 입력하세요. 10
KeyboardInterrupt
```

2 입력 4 입력 6 입력

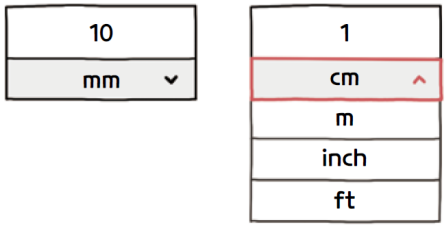
프로그램을 중단하고 싶을 때는 Ctrl + C 를 누르고 터틀 그래픽 창을 닫습니다.

실전 예제

실전 예제1 – 단위 환산 프로그램

문제

mm 단위의 길이를 입력하면 cm, m, inch, ft 등으로 단위가 변환되어 출력되는 함수가 포함된 프로그램을 만들어 봅시다.




```
길이(mm)를 입력하세요. 1
1 mm -->  0.1 cm
1 mm -->  0.001 m
1 mm -->  0.03937 inch
1 mm -->  0.003281 ft
```

```
길이(mm)를 입력하세요. 55
55 mm -->  5.5 cm
55 mm -->  0.055 m
55 mm -->  2.16535 inch
55 mm -->  0.180455 ft
```

실전 예제1 – 단위 환산 프로그램

해결

ch11_appEx_01.py

```
01  def convertUnit(lenMm):
02
03      unitDic = {}
04
05      unitDic['cm'] = lenMm * 0.1
06      unitDic['m'] = lenMm * 0.001
07      unitDic['inch'] = lenMm * 0.03937
08      unitDic['ft'] = lenMm * 0.003281
09
10      return unitDic
11
12  def printLength(lengths):
13      for len in lengths.keys():
14          print(inputData, 'mm --> ', lengths[len], len)
15
16  inputData = int(input('길이(mm)를 입력하세요. '))
17
18  result = convertUnit(inputData)
19  printLength(result)
```

실전 예제2 – 할인된 상품 가격표 출력 프로그램

문제

한빛 마트는 고객 감사의 일환으로 '오늘의 할인' 이벤트를 진행할 계획입니다. 아래의 상품 가격표를 참고해서 '오늘의 할인율'을 입력하면 할인된 가격이 출력되는 프로그램을 만들어 봅시다.

상품	쌀	상추	고추	마늘	통닭	햄	치즈
가격	9,900	1,900	2,900	8,900	5,600	6,900	3,900


```
-------------------------------------------
-- 한빛마트 오늘의 할인 가격표 출력 시스템 --
-------------------------------------------
오늘의 할인율을 입력하세요. 15
쌀    : 9900 원 15 %DC -> 8415 원
상추 : 1900 원 15 %DC -> 1615 원
고추 : 2900 원 15 %DC -> 2465 원
마늘 : 8900 원 15 %DC -> 7565 원
통닭 : 5600 원 15 %DC -> 4760 원
햄    : 6900 원 15 %DC -> 5865 원
치즈 : 3900 원 15 %DC -> 3315 원
-------------------------------------------
```

실전 예제2 – 할인된 상품 가격표 출력 프로그램

해결

ch11_appEx_02.py

```
01  standardPrice = {'쌀':9900, '상추':1900,'고추':2900,'마늘':8900,'통닭':5600,
                     '햄':6900, '치즈':3900}
02
03  def getDiscountPrice(rate):
04      dcPrice = {}
05      for goods in standardPrice.keys():
06          disPri = int(standardPrice[goods] * (1 - (rate / 100)))
07          dcPrice[goods] = disPri
08      return dcPrice
09
10  def printPrice(priceList):
11      for goods in priceList.keys():
12          print(goods, '\t:', standardPrice[goods], '원\t', inputData, '%DC ->',
                  priceList[goods], '원')
13
14  print('-----------------------------------------')
15  print('-- 한빛마트 오늘의 할인 가격표 출력 시스템 --')
16  print('-----------------------------------------')
17
18  inputData = int(input('오늘의 할인율을 입력하세요. '))
19  discountPrice = getDiscountPrice(inputData)
20
21  printPrice(discountPrice)
22  print('-----------------------------------------')
```

Thank you!

한빛아카데미 Hanbit Academy, Inc.