Exercise

- Write a function that *gets 2 numbers from the user* and passes them both back (pass by reference).
  - What are the input values? What type of parameters should they be?
  - What are the output values? Are they parameters or local variables?
- Write a function that takes one integer as input and gets another number from the user. This function must pass back the sum (+) and difference (-) of the two numbers.
  - What are the input values? What type of parameters should they be?
  - What are the output values? Are they parameters or local variables?

Function input and output

- Functions are characterized in terms of:
  - Input (data that must be passed in to the function)
  - Output (data the function passes back to the caller)
  - Side-effects (other changes, like printing or waiting for user input)
- Data can only be stored in three types of variables:
  - Value parameters
  - Reference parameters
  - Local variables

Function input and output

- Parameters support input
  - Only reference parameters also support output
  - Local variables can not be used for input
  - A single parameter or variable can be output with `return`

- Data requirements dictate parameters
  - Input only: value parameter
  - Output only:
    - Local variable and return (if only 1 output)
    - Reference parameter (if more than 1)
  - Input and output: reference parameter

Exercise

- Consider the following function body
  - Write an appropriate heading for this function
    - Choose any name you like for the function
    - What parameters? Reference or value?
      - (Assume a, b and c are all *int*)
      - Assume to update the values of all parameters.

```
{
  c = (a * b)/2;
  if( a > b )
  {
   b = 17;
  }
  else
  {
   b = 20;
  }
```

Example: Largest and Smallest

1. Get 4 numbers from the user
2. Print the largest and smallest of those numbers

//Bubble Sort (1st algorithm we would learn in CS). Sorry. This is for sorting the numbers in increasing or decreasing orders.

largest and smallest of those numbers

Numbers: 25, 87, 65, 15, 7, 100, -10 (min: 7; max: 100)

Find min:

mini_variable = 10000; max_variable = 0;

// mini_variable = positive_Infinity; max_variable = negative_infinity;

mini_variable (10000) compare with the rest (25, 87, 65, 15, 7, 100)

10000 v.s. 25: if 10000 > 25: update

update the mini_variable = 25;

25 v.s. 87: if 25 > 87: swap; otherwise: do not do anything;

25 v.s. 65: if 25 > 65: swap; otherwise: do not do anything;

25 v.s. 15: if 25 > 15: update; otherwise: do not do anything;

update the mini_variable = 15;

15 v.s. 7: if 15 > 7: update; otherwise: do not do anything;

update the mini_variable = 7;

7 v.s. 100: if 7 > 100: update; otherwise: do not do anything;

7 v.s. -10: if 7 > -10: update; otherwise: do not do anything;

update the mini_variable = -10;

// after the last update, the mini_variable value is 7.

return mini_variable

Example: Largest and Smallest

1. Get any number of numbers from the user
   - Stop when the user types in -1
2. Print the largest and smallest of those numbers

- How would you do it?
  - Imagine a deck of cards with numbers on them
  - How would you find the largest and smallest numbers in the deck?

Example: Largest and Smallest

1. Get any number of numbers from the user
   - Stop when the user types in -1 // while loop in the function
2. Pass back the largest and smallest of those numbers

largest, smallest

- Now make step 1 a function
  - What are the input parameters?
  - What are the values you want to output?
    - Return can only return a single value
    - Use reference parameters to output more than one value

Largest = -10000000;

Smallest = 10000000;

Num = 14

Lines 38-39: ? 14> -10000000 (T), largest = 14;

? 14 < 10000000 (T), smallest = 14;

Num = -100

Lines 38-39: ? -100 > 14 (F) // largest is kept as 14

? -100 < 14 (T), smallest = -100;

// largest = 14; smallest = -100;