Additional practice problems for Week 1

These are optional additional practice problems. For each problem create a directory with the problem name and inside that folder create a C file with the same name.

w1ap1-factorial

Write a function to calculate the factorial of a given number. The function takes one parameter, an integer number for which to calculate the factorial, and returns the result as an integer. The function must return -1 if factorial can not be calculated. Write a main function to test the previously defined function.

Hint

What is the largest number for which your function can produce a correct answer?

Example

```
w1ap1-factorial/ $ ./w1ap1-factorial
Factorial(n). n=? 5
Factorial of 5 is 120
```

Testing and submitting your code

```
check50 okskola/cs50addp/main/w1ap1-factorial
submit50 okskola/cs50addp/main/w1ap1-factorial
```

w1ap2-digisum

Write a function that, given an positive integer n, returns the sum of its first and last digits. Return -1 if n is incorrect. Write a main function to test the previously defined function.

Hint

The modulo operator (%) and integer division can help to extract individual digits.

Example

```
w1ap2-digisum/ $ ./w1ap2-digisum
n=? 135
Sum of first and last digits of n is 6
w1ap2-digisum/ $ ./w1ap2-digisum
n=? 6
Sum of first and last digits of n is 12
```

Testing and submitting your code

```
check50 okskola/cs50addp/main/w1ap2-digisum
submit50 okskola/cs50addp/main/w1ap2-digisum
```

w1ap3-reverse

Write a function to reverse a given positive integer. The function should return the reversed value, where all digits are in the reversed order. If the number is negative, then return the number itself. Write a main function to test the previously defined function.

Example

```
w1ap3-reverse/ $ ./w1ap3-reverse
```

```
n=? 12345
Reversed n is 54321
w1ap3-reverse/ $ ./w1ap3-reverse
n=? -12345
Reversed n is -12345
```

Testing and submitting your code

```
check50 okskola/cs50addp/main/w1ap3-reverse
submit50 okskola/cs50addp/main/w1ap3-reverse
```

w1ap4-sumsum

Write a function sum(n) which returns sum of all integers from 1 to n. Then write a function sumsum(n,t) which applies function sum() t times: the first time to the original n, and each subsequent time to the result of the previous sum(). Write a main function which lets the user to enter n and t and checks that n≥1 and t≥1.

Hint

Do we always need a loop to calculate something?

Example

```
w1ap4-sumsum/ $ ./w1ap4-sumsum
n: 4
t: 1
Sumsum: 10
w1ap4-sumsum/ $ ./w1ap4-sumsum
n: 3
t: 2
Sumsum: 21
```

Testing and submitting your code

```
check50 okskola/cs50addp/main/w1ap4-sumsum
submit50 okskola/cs50addp/main/w1ap4-sumsum
```