```
# CMPT 145 Course material
# Copyright (c) 2017-2021 Michael C Horsch
# All rights reserved.
#
# This document contains resources for homework assigned to students of
# CMPT 145 and shall not be distributed without permission.  Posting this
# file to a public or private website, or providing this file to a person
# not registered in CMPT 145, constitutes Academic Misconduct, according
# to the University of Saskatchewan Policy on Academic Misconduct.
#
# Synopsis:
#    Assignment 7 solutions


def fibonacci(n):
    """
    Purpose:
        The Fibonacci numbers are: 0, 1, 1, 2, 3, 5, 8, 13, ...
    Preconditions:
        :param n: a non-negative integer
    Return:
        :return: the nth Fibonacci number, starting with fib(0) = 0
    """
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return fibonacci(n-1) + fibonacci(n-2)


def moosonacci(n):
    """
    Purpose:
        Calculate the nth Moosonacci number
        The Moosonacci numbers are: 0, 1, 2, 3, 6, 11, 20, 37, ...
    Preconditions:
        :param n: a non-negative integer
    Return:
        :return: the nth Moosonacci number, starting with moos(0) = 0
    """
    if n == 0:
        return 0
    elif n == 1:
        return 1
    elif n == 2:
        return 2
    else:
        return moosonacci(n-1) + moosonacci(n-2) + moosonacci(n-3)


def recsum(i, j):
    """
    Purpose:
        Calculate the sum of integers i through j (exclusive)
    Preconditions:
        :param i: a non-negative integer
        :param j: a non-negative integer
    Return:
```

```
        :return: the sum i + (i+1) + ... + (j - 1)
    """
    if i >= j:
        return 0
    else:
        return i + recsum(i + 1, j)



def countoddrec(i, j):
    """
    Purpose:
        Calculate the number of odd integers in the
        range starting at i through j (exclusive)
    Preconditions:
        :param i: a non-negative integer
        :param j: a non-negative integer
    Return:
        :return: the number of odd integers in the range
    """
    if i >= j:
        return 0
    elif i % 2 == 1:
        return 1 + countoddrec(i + 2, j)
    else:
        return countoddrec(i + 1, j)
```