Introduction to Computer Science Module: CH-232
Constructor University Date: 2023-09-15
Dr. Jürgen Schönwälder Due: 2023-09-22

Problem Sheet #2

Problem 2.1: *proof by contrapositive* (4 points)

Let $a \in \mathbb{Z}$ be an integer number. If a^{32} is an odd number, then a^4 is an odd number as well.

Problem 2.2: *proof by induction* (4 points)

Let $n \in \mathbb{N}$ be a natural number with $n \geq 1$. Prove by induction that $n^3 + (n+1)^3 + (n+2)^3$ is divisible by 9.

Problem 2.3: *sum of divisors in haskell* (1+1 = 2 points)

The sum of divisors function $\sigma_z(n)$ is defined as the sum over all divisors of a number n taken to the power of z. The function $\sigma_z(n)$ can be more formally defined as

$$\sigma_z(n) = \sum_{d|n} d^z$$

where $d|n$ is a shorthand for "d divides n". We implement this function in two steps.

a) Write a function `divisors :: Int -> [Int]` returning the list of divisors of a given positive integer `n`. The list of divisors includes 1 and the number `n` itself. Some examples:

```
ghci > divisors 1
[1]
ghci > divisors 6
[1,2,3,6]
ghci> divisors 12
[1,2,3,4,6,12]
ghci > divisors 15
[1,3,5,15]
ghci > divisors 16
[1,2,4,8,16]
```

Consider to define your function using a list comprehension. Here is a template to get started. Replace `undefined` with a suitable list comprehension.

```haskell
-- Return the list of positive divisors of an integer n.
divisors :: Int -> [Int]
divisors n = undefined
```

Recall that the Haskell function `div` gives you the result of an integer division (truncated toward negative infinity) and the function `mod` gives you the integer modulus (remainder of an integer division).

b) Write a function `sigma :: Int -> Int -> Int` that takes the two arguments z and n and returns the sum of the zth powers of the positive divisors of n. You can use the `sum` function to calculate the sum of a list of numbers. Here is a template to get started. Replace `undefined` with a suitable list comprehension.

```haskell
-- Return the sum of divisors of n taken to the power of z
sigma :: Int -> Int -> Int
sigma z n = sum undefined
```

Some sample results:

```
ghci > sigma 0 1
1
ghci > sigma 0 12
6
ghci > sigma 1 12
28
```