Algorithms and Datastructures assignment 1

Thomas Broby Nielsen (xlq119)
Tobias Overgaard (vqg954)
Christian Buchter (zvc154)

14. maj 2015

Indhold

1 Task 1

$$|N(c,i)| = \begin{cases} 0 & \text{if } c \leq 0 \\ 0 & \text{if } i = 0 \\ 1 + N(c, i-1) & \text{if } p_i = c \\ N(c - p_i, i-1) + N(c, i-1) & \text{otherwise} \end{cases}$$

2 task 2

In order to prove the correctness of our formula, for calculating the amount of unique combinations of unique beers you can buy with a specified amount of money, we will have to tjeck the three properties of a correct formula.

First we will first prove the correctness of the initialization of our formula.
This we have summarized into showing that for a given value of $1 \leq c$ and $1 \leq i$ that our base case holds true for the first iteration. For the example above, it is trivial to see that if $N(1,1)$ then the base cases for the formula, won't be triggered. From this we can see that initialization will hold for any value of $1 \leq c$ and $1 \leq i$.

The second property that our formula must hold, is that if the invariant before an iteration is true, must remain true after the iteration. To show that our

To further prove the correctness of our formula, we have see if the formula holds true before and after an iteration.

3 Task 3

Bellow we have written the pseudo-code for MemoizedBeerComp using the DP-method

```
MemoizedBeerComp(c, i, beerList, array)
1 if c <= 0 or i==0
2    return 0
3 if array(c, i) > -1
4    return array(c, i)
```

```
if c - beerList[i] == 0
    array(c,i) = 1 + MemoizedBeerComp(c, i - 1)
    return array(c,i)
  else
       array(c,i) = MemoizedBeerComp(c, i - 1, beerList, array) +
                    MemoizedBeerComp(c - beerList[i], i - 1, beerLis, array)
       return array(c, i)
```

Because we coulden't get our heads around to calculate the running time in task 4 for our top-down implementation of memoizedBeerComp(until after we talked with a TA), we went ahead and also implemented our formula as a bottoup method as seen below.

```
Bottom-Up-Beers (P,c) =
    let r[1,2, ... ,c][0,1,2,...,P.length] be a new Matrix
    for j=1 to c{
       r[j][0] = 0;
       for k = 1 to P.length {
          q = 0;
          if (P[k] == j) {q+=1}
          q += r[j][k-1];
          if (j-P[k] > 0) {
            q += r[j-P[k]][k-1];
           }
           r[j][k] = q;
        }
    }
   return r[c][P.length];
```

4 Task 4