Name:

Algorithms and Data Structures
Jacobs University Bremen
Dr. Florian Rabe

Quiz 6
given: 2017-05-04

You have 20 minutes.

Problem 1

Points: 2+3+5

Consider the following recursive function:

```
fun additiveFold(x : IndList[ℤ]) : ℤ =
  match x
    Nil ↦ 0
    Cons(hd, tl) ↦ hd + additiveFold(tl)
```

1. Give the result of $additiveFold([1,2,3,4])$.
2. Explain why this function is not tail-recursive.
3. Convert it to a tail-recursive function by completing the dotted parts below.

```
fun additiveFoldAux(x : IndList[ℤ], result : ℤ) : ℤ =
  match x
    Nil ↦ ......
    Cons(hd, tl) ↦ ......
fun additiveFold(x : IndList[ℤ]) : ℤ = {additiveFoldAux(...... , ......)}
```

Problem 2

Points: 5+3+2

Assume an unlabeled directed graph G with distinguished nodes *start* and *end*.

```
fun search(state : List[Node]) : Option[List[Node]] =

  if (abort(state)) {return None}
  if (solution(state)) {return Some(state)}
  foreach(choices(state), c ↦
    x := search(state + [c])
    if (x ≠ None) {return x}
  )
  return None

fun choices(state : List[Node]) : List[Node] =
  outgoing(G, last(state))

fun solution(state : List[Node]) : bool =
  last(state) == end

fun abort(state : List[Node]) : bool =
  contains(init(state), last(state))

fun init[A](x : List[A]) : List[A] =
  x without its last element
fun last[A](x : List[A]) : A =
  the last element of x
```

Consider the backtracking algorithm on the left.

1. What does $search([start])$ return?
2. What is the purpose of the function *abort* in general?
3. What is the purpose of the function *abort* in this particular case?