Abigail Rechkin
Alex Yavnilovitch

CS 577 – HOMEWORK 2

ABIGAIL RECHKIN
ALEXANDRA YAVNILOVITCH

Algorithm 3:

Input: A list A of n positive integers where $n \geq 0$

Output: The number of students who can see the canteen.

```
1: Procedure WHOCANSee(A[1,....,n])
2: return Foo(A[1.,...,n], 0)
```

Input: An array A of positive integers of length $n \geq 0$.
An integer $t \geq 0$

Output: The number of students who can see the canteen.

```
3:  Procedure Foo(A[1,...,n], t)
4:      if n=0 then
5:          return 0
6:      else
7:          h ← A[1]
8:          if h>t then
9:              return 1 + Foo(A[2,...n], h)
10:         else
11:             return Foo(A[2,....,n], t)
```

PROOF

Let $A[1,...,n]$ and t be valid inputs to Foo, and consider the complexity measure $\mu(A[1,...,n],t) = n$. For A to be valid, $n \geq 0$, and hence $\mu(A[1,...,n],t) \geq 0$. Now we are going to use structural induction, to prove that for all positive integers m, the program is correct on all valid inputs A, t with $\mu(A,t) \leq m$.

BASE CASE $m=0$

Since $n=0$, the line contains 0 students, and on line 5, the algorithm returns 0. This is the correct output, because 0 students in the given input array can see the canteen.
(*) □

INDUCTIVE STEP

INDUCTIVE HYPOTHESIS: We assume that for some integer $n \geq 1$, the algorithm is correct for valid inputs A' and t', with $\mu(A',t') \leq n-1$.

Now, consider the valid input A and t with $\mu(A,t) = n$.

(*) Input t represents the maximum height seen in the entire array A. If $t=0$, we conclude that there are no students in line (Student heights must be positive integers).

CASE 1 $h > t$

On line 7, h is assigned $A[1]$. Since the first Student's height h is greater than the current maximum height, we may conclude that the Student can see the canteen. On line 9 we make a recursive call with inputs $A[2,\ldots,n]$ and h. Since $h > t$, and $t \geq 0$ based on our input specification, $h \geq 0$ and is a valid input. Since length of A is $n-1$, and $n > 0$ and an integer, we know that $n-1 \geq 0$, and is an integer. Thus list $A[2,\ldots n]$ is a valid input, and $\mu(A[2,\ldots,n],h) = n-1$ which makes $\mu(A[2,\ldots,n],h) < n$. By the I.H., the algorithm correctly returns the number of students who can see the canteen, in the subset of $A[2,\ldots,n]$. This value is added with 1 to account for $A[1]$, which sums to the correct return value.

CASE 2 $h \leq t$

Since the first student's height $A[1]$ is less than the current maximum height t, we know that the Student cannot see the canteen. On line 11, we make a recursive call with inputs $A[2,\ldots,n]$, and t. Since t remains unchanged, and based on input specification $t \geq 0$ and is an integer, we conclude that t is a valid input. Since the length of input A is $n-1$, and we know that $n \geq 1$ (base case not entered), we may conclude that $n-1 \geq 0$, and is an integer. Thus, the input list A is of valid length $n-1$, and is a valid input. The measure of complexity $\mu(A[2,\ldots,n],t) = n-1$ at this point, which is less than $\mu(A[1,\ldots,n],t) = n$. Because $\mu(A[2,\ldots,n],t) < \mu(A[1,\ldots,n])$, we observe that the complexity measure decreases on each recursive step. By I.H., we know that $Foo(A[2,\ldots,n],t)$ will return the number of remaining students in subset A, who can see the canteen. Furthermore, since the first student cannot see the canteen, this is the correct output.

Hence, by induction, the algorithm is correct on all valid inputs A and t. ∎