***(\* Proposed solution to Problem 1b, Assignment II. Assumes “while” and
 “assignment” expressions, upon completion, evaluate to unit values. \*)***
fun P(A:(int->int)array ref array): int ref array ref =
 ***(\* initialize three refs:
 i indexes main array A, j indexes subarrays of A, and
 count counts how many elements our return array will need \*)***
 let i=ref 0 in let j=ref 0 in let count=ref 0 in
 try { ***(\* iterate through A as long as it has more elements \*)*** while(true) {
 A[!i]; ***(\* make sure we haven’t run off the end of A \*)***
 try { ***(\* iterate through one of A’s subarrays \*)*** while(true) {
 ***(\* if appropriate, increment count \*)*** if 0 < (!((!!(A[!i]))[!j]))(0) then count:=!count+1 else ();
 j:=!j+1 ***(\* goto next element in subarray \*)***
 }
 } with j:=0; i:=!i+1 ***(\* goto next element in A \*)***
 }
 } with ();
 ***(\* now we can initialize the return array B \*)*** let B = arr[!count].init(ref 0) in
 count:=0; i:=0; j:=0; ***(\* reset the counter and array-index iterators \*)***
 try { ***(\* again iterate through A as long as it has more elements \*)***
 while(true) {
 A[!i]; ***(\* make sure we haven’t run off the end of A \*)*** try { ***(\* iterate through one of A’s subarrays \*)***
 while(true) {
 ***(\* if appropriate, set a value in B and then increment count\*)***
 let k = (!((!!(A[!i]))[!j]))(0) in
 if 0<k then B[!count]:=ref k; count:=!count+1 else ();
 j:=!j+1 ***(\* goto next element in subarray \*)***
 end
 }
 } with j:=0; i:=!i+1 ***(\* goto next element in A \*)***
 }
 } with ();
 ***(\* finished setting values in B, so return a pointer to it \*)***
 ref B
 end
 end end end