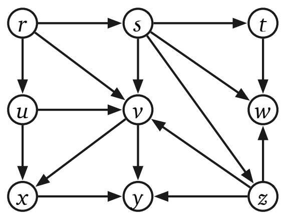
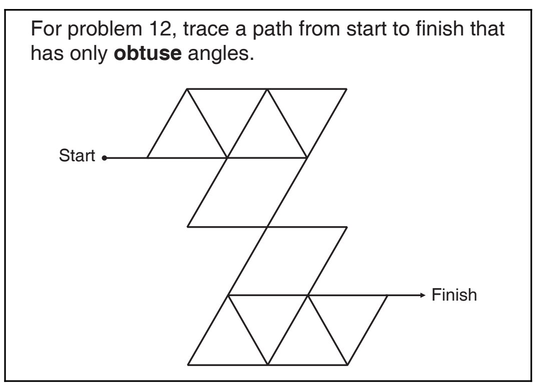
Write your answers in the separate answer booklet.
Please return this question sheet and your cheat sheet with your answers.

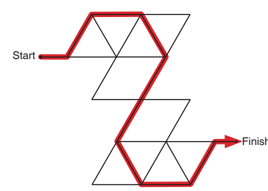
1. ***Clearly*** indicate the following structures in the directed graph below, or write NONE if the indicated structure does not exist. (There are several copies of the graph in the answer booklet.)



   (a) A depth-first spanning tree rooted at r.
   (b) A breadth-first spanning tree rooted at r.
   (c) A topological order.
   (d) The strongly connected components.

2. The following puzzle appears in my younger daughter's math workbook.[1] (I've put the solution on the right so you don't waste time solving it during the exam.)





   Describe and analyze an algorithm to solve arbitrary obtuse-angle mazes.

   You are given a connected undirected graph G, whose vertices are points in the plane and whose edges are line segments. Edges do not intersect, except at their endpoints. For example, a drawing of the letter X would have five vertices and four edges; the maze above has 17 vertices and 26 edges. You are also given two vertices Start and Finish.

   Your algorithm should return True if G contains a walk from Start to Finish that has only obtuse angles, and False otherwise. Formally, a walk through G is valid if $90^\circ < \angle uvw \leq 180^\circ$ for every pair of consecutive edges $u{\to}v{\to}w$ in the walk. Assume you have a subroutine that can compute the angle between any two segments in $O(1)$ time. Do ***not*** assume that angles are multiples of 1°.

[1] Jason Batterson and Shannon Rogers, *Beast Academy Math: Practice 3A*, 2012. See https://www.beastacademy.com/resources/printables.php for more examples.

3. Suppose you are given two unsorted arrays $A[1..n]$ and $B[1..n]$ containing $2n$ distinct integers, such that $A[1] < B[1]$ and $A[n] > B[n]$. Describe and analyze an efficient algorithm to compute an index i such that $A[i] < B[i]$ and $A[i+1] > B[i+1]$. *[Hint: Why does such an index i always exist?]*

4. You have a collection of n lockboxes and m gold keys. Each key unlocks *at most* one box; however, each box might be unlocked by one key, by multiple keys, or by no keys at all. There are only two ways to open each box once it is locked: Unlock it properly (which requires having a matching key in your hand), or smash it to bits with a hammer.

   Your baby brother, who loves playing with shiny objects, has somehow managed to lock all your keys inside the boxes! Luckily, your home security system recorded everything, so you know which keys (if any) are inside each box. You need to get all the keys back out of the boxes, because they are made of gold. Clearly you have to smash at least one box.

   (a) Your baby brother has found the hammer and is eagerly eyeing one of the boxes. Describe and analyze an algorithm to determine if it is possible to retrieve all the keys without smashing any box except the one your brother has chosen.

   (b) Describe and analyze an algorithm to compute the minimum number of boxes that must be smashed to retrieve all the keys.

5. A *shuffle* of two strings X and Y is formed by interspersing the characters into a new string, keeping the characters of X and Y in the same order. For example, the string BANANAANANAS is a shuffle of the strings BANANA and ANANAS in several different ways.

   BANANA ANANAS   BAN ANA ANA NAS   B AN AN A A NA NA S

   Given three strings $A[1..m]$, $B[1..n]$, and $C[1..m+n]$, describe and analyze an algorithm to determine whether C is a shuffle of A and B.