

Midterm Exam

Name: _____

SS Number: _____

Instructions: Answer all questions.

1. (25) Consider the set $O = \{o_1, o_2, \dots, o_n\}$ of n objects of any kind. In how many different ways can you pair two different objects, as a function of n , if pairings are unordered (if pairings are unordered then pairing $\{o_i, o_j\}$ is considered the same as pairing $\{o_j, o_i\}$, regardless of the indices i and j)? Prove your answer is correct (remember that $T(n)$ stuff?).

Answer: $n(n-1)/2$

Proof: Let $T(n)$ be the number of pairings for n objects. Then construct the recurrence:

$$T(2) = 1$$

$$T(n) = T(n-1) + n - 1 \quad \text{Because there are } n - 1 \text{ connections to the } n^{\text{th}} \text{ object.}$$

Solving by adding equations gives $T(n) = 1 + 2 + \dots + (n-1)$. Pairing $n-1$ with 1 to get a sum equal to n , $n-2$ with 2 to get another n , $n-3$ with 3 to get another n , and so on, we get $(n-1)/2$ of these n 's. Hence $T(n) = n(n-1)/2$.

2. (10) Call two objects o_i and o_j *connected* by a set S of unordered pairings if S contains a subset that can be organized as follows

$$\{o_i, o_{\pi_1}\}, \{o_{\pi_1}, o_{\pi_2}\}, \{o_{\pi_2}, o_{\pi_3}\}, \dots, \{o_{\pi_{x-1}}, o_{\pi_x}\}, \{o_{\pi_x}, o_j\}$$

for some x and some mapping π of indices, where it is perfectly OK to reverse one or more pairings to achieve this (since the pairings are unordered). In other words, the usual notion of being connected if pairings are edges without arrows. Consider the following algorithm:

Put all pairings of objects of O in a red box.

Open a second completely empty black box.

Repeat the following until the red box is empty:

Arbitrarily remove a pairing $\{o_i, o_j\}$ from the red box.

If o_i and o_j had been but are no longer connected by the pairings in the red box then put $\{o_i, o_j\}$ into the black box, otherwise throw the pairing out.

end of repeat

How many pairings are in each box after the algorithm terminates, as a function of n ?

Red box: 0

Black box: $n - 1$

3. (5) What is so special about the objects of O with respect to the pairings of the black box after the algorithm terminates?

They are all connected by the pairings in the black box and removing any pairing in the black box causes at least two objects not to be connected any more.

4. (15) What makes you think so?

- (a) Any pairwise disconnected set O' of objects in the red box is pairwise connected in the black box. The only way for a set O' to become pairwise disconnected is if a removed pairing p contains an object in one pairwise disconnected subset of O' and a second object in the complementary subset of O' . If the two subsets are each pairwise connected in the black box, that pairing (p), which is now added to the black box, connects all objects of both subsets and therefore O' . In the base case the subsets are one object each and connection of these two objects in the black box is directly due to the pairing (p) being placed there.
- (b) All objects become pairwise disconnected in the red box because all pairings eventually are removed from the red box.
- (c) Therefore, from 4a and 4b, *all objects are pairwise connected in the black box at the end of the algorithm.*
- (d) Every pairing p added to the black box causes two separately pairwise connected sets of objects to become a single pairwise connected set. Otherwise, in case of no change, from 4a, there would be no change in pairwise disconnected sets in the red box and the pairing (p) would have been thrown out. The case of three or more separately pairwise connected sets joining to become a single pairwise connected set is impossible since a pairing can only affect at most two subsets at a time.
- (e) Since the number of mutually connected sets in the black box is initially n , from 4d and 4c the number of pairings added to the black box is $n - 1$.
- (f) If a pairing can be removed from the black box without affecting connectedness then, from 4d, another $n - 1$ pairings can be removed to pairwise disconnect all objects. But that implies there were n pairings in the black box upon termination of the algorithm, in violation of 4e. Hence *removing any pairing from the black box causes at least two objects not to be connected.*

5. (5) What C++ object(s) would you use to represent the red box (provide local state and descriptive method names)?

An object of class `RedBox` needs to efficiently allow insertion and removal of any kind of objects. It also must also efficiently implement find and disconnection operations on sets. This can be supported using an adjacency list representation of the pairings. Then the `bool find (void*, void*)` method can be built to take two objects of a pairing as input and return true if the objects are connected and false otherwise. The `find` method can do a depth-first search on the adjacency list structure beginning at one of the two input objects. The `void *remove (void*)` method can be used to remove an object from the two adjacency lists involving the 'found' objects.

6. (5) What C++ object(s) would you use to represent the black box (provide local state and method names)?

An object of any container class allowing efficient insertion and the ability to efficiently read the contents of the container without destroying the organization of the objects in the container, for example a `List` class object. Assuming a pointer to the `List` class is hidden as a private variable, public methods `void insert(Pairing *)`, `void readList()`, and `bool end_of_list()` would suffice.

7. (5) What C++ object(s) would you use to represent the pairings (local state and descriptive method names)?

The `Pairing` class (used above) should have two `void *` pointers to hold two of any kind of object. Assuming they are private, public methods `void *left_obj()` and `void *right_obj` would be enough to identify objects so they can be checked for connectedness. A pairing would be established using a constructor of two `void *` arguments. Use adjacency lists to store the pairings.

8. (30) Describe how you would implement the check called for in the `If` statement of the algorithm (there are lots of possible correct answers to this - you get more points for more efficient proposals).

Using an adjacency list representation for the pairings, perform a depth-first search through the structure to determine whether two objects are connected. Modify the structure by removing the entries corresponding to the two objects.