How to Design (Iterative) Algorithms

- establish the problem
- identify avenues of attack
- define the algorithm
- show termination and correctness
- determine efficiency

How to Design (Iterative) Algorithms

Identify avenues of attack.

- *Paradigms and patterns.*
  Consider the iterative patterns defined in section 3.1.

pattern	loop structure
process input	for each input element, process that element and incorporate it into the solution so far
produce output	repeatedly produce the next output element repeatedly produce the next piece of the solution
narrow the search space	repeatedly eliminate some non-solutions

How to Design (Iterative) Algorithms

Define the algorithm. The core of an iterative algorithm is defining the loop.

- *Main steps.*
  This is the core of the algorithm — the loop body. What's being repeated?
- *Exit condition.*
  When does the loop end?

pattern	loop structure	exit condition
process input	for each input element, process that element and incorporate it into the solution so far	when all of the input elements have been processed
produce output	repeatedly produce the next output element repeatedly produce the next piece of the solution	when all of the output elements have been produced when the solution is complete
narrow the search space	repeatedly eliminate some non-solutions	when the solution has found or there are no solutions left

- *Setup.*
  Whatever must happen before the loop begins.
- *Wrapup.*
  Whatever must happen to get the final answer after the loop ends.
- *Special cases.*
  Make sure the algorithm works for all legal inputs — identify the cases that need to be handled and address how that handling is incorporated into the previous steps (if not already accounted for).
- *Algorithm.*
  Assemble the algorithm from the previous steps and state it.

  There shouldn't be new elements here, instead bring together the main steps, exit condition, setup, and wrapup along with any handling needed for special cases and state the whole algorithm.

In a group of people, it is to be expected that some of them may not want to work with each other. Assuming that each person has at most d other people that they don't want to work with, divide the people into d+1 groups so that everyone is in exactly one group and no one is in a group with someone they don't want to work with.

pattern	loop structure	exit condition
process input	for each input element, process that element and incorporate it into the solution so far	when all of the input elements have been processed

Main steps.

incorporate the next input item into the solution to obtain a solution for one more element	3 respondents	60 %
for each element, process it	3 respondents	60 %
for each person, put them into a group	2 respondents	40 %
for each person, put them into the first group not containing someone they don't want to work with	5 respondents	**100 %** ✓
produce the next output item		0 %
for each group, add people to that group as long as there isn't anyone in the group that they don't want to work with	3 respondents	60 %

Exit condition. The loop ends when --

all of the input items have been processed	3 respondents	60 %
when all of the ouput items have been produced	1 respondent	20 %
when every person has been added to a group	3 respondents	**60 %** ✓
when everyone has been assigned to a legal group	5 respondents	100 %
when no one else can be added to the current group		0 %
when all of the groups are full		0 %
when d+1 groups have been formed	1 respondent	20 %

How to Design (Iterative) Algorithms

Show termination and correctness. Show that the algorithm produces a correct solution.

- *Termination.* Show that the loop — and thus the algorithm — always terminates.
  - *Measure of progress.*
    Identify a quantity and the direction of change that leads towards the exit condition.
  - *Making progress.*
    Explain why every iteration of the loop advances the measure of progress towards the exit condition.
  - *The end is reached.*
    Explain why making progress ensures that the exit condition is always reached.

pattern	measure of progress	making progress	termination argument
process input	number of input elements processed	each iteration processes one more element	repeatedly processing one more input element means that eventually all will have been processed
produce output	number of elements in the solution	each iteration produces one more element	repeatedly producing one more output element or one more piece of the solution means that eventually all will have been produced
narrow the search space	size of the current range or (alternatively) the number of solutions still in the current range	each iteration reduces the size of the search space	repeatedly reducing the size of the current range or the number of solutions still in the current range means that eventually there will be no solutions left if the solution hasn't been found

In a group of people, it is to be expected that some of them may not want to work with each other. Assuming that each person has at most d other people that they don't want to work with, divide the people into d+1 groups so that everyone is in exactly one group and no one is in a group with someone they don't want to work with.

pattern	measure of progress	making progress	termination argument
process input	number of input elements processed	each iteration processes one more element	repeatedly processing one more input element means that eventually all will have been processed

Exit condition. The loop ends when --

all of the input items have been processed	3 respondents	60 %
when all of the ouput items have been produced	1 respondent	20 %
when every person has been added to a group	3 respondents	**60 %** ✓

Measure of progress.

something to count the number of people added to groups	2 respondents	40 %
the number of elements considered	2 respondents	40 %
the number of elements left to consider	1 respondent	20 %
number of input items processed	4 respondents	80 %
number of output items processed		0 %
number of groups that have been filled	1 respondent	20 %
number of people that have been assigned to a group	3 respondents	**60 %** ✓
number of people not yet assigned to a group	5 respondents	100 %

How to Design (Iterative) Algorithms

- *Correctness.* Show that the algorithm is correct.
  - *Loop invariant.*
    State a loop invariant.
  - *Establish the loop invariant.*
    Explain why the loop invariant holds at the beginning of the first and second iterations of the loop.
  - *Maintain the loop invariant.*
    Explain why the loop invariant continues to be true after each iteration — assuming that it holds at the beginning of iteration k, explain why it also holds at the beginning of the next iteration $(k+1)$.
  - *Final answer.*
    Explain why the whole algorithm — setup, loop, wrapup — means that the final result is a correct answer to the problem.

pattern	loop invariant
process input	have a correct solution for the first k input elements, or (alternatively) haven't gone wrong yet (solution so far is consistent with a solution for the whole problem)
produce output	have produced the first k elements of the correct output
narrow the search space	either the element is within the current search space / set of solutions or it was never present at all, or (alternatively) not all of the solutions (if there are any) have been eliminated

In a group of people, it is to be expected that some of them may not want to work with each other. Assuming that each person has at most d other people that they don't want to work with, divide the people into d+1 groups so that everyone is in exactly one group and no one is in a group with someone they don't want to work with.

Main steps.

for each person, put them into the first group not containing someone they don't want to work with	5 respondents	**100 %** ✓

Exit condition. The loop ends when --

when every person has been added to a group	3 respondents	**60 %** ✓

pattern	loop invariant
process input	have a correct solution for the first k input elements, or (alternatively) haven't gone wrong yet (solution so far is consistent with a solution for the whole problem)

Loop invariant.

the first k people have been assigned to groups	2 respondents	40 %
the first k people have been assigned to groups so that no one is in a group with someone they don't want to work with	4 respondents	80 %
there are at most d+1 groups	2 respondents	40 %
the first k people have been assigned to groups so that no one is in a group with someone they don't want to work with and there are at most d+1 groups	4 respondents	**80 %** ✓
have a correct solution for the first k input items	2 respondents	40 %
haven't gone wrong yet	1 respondent	20 %
have produced the first k items of the output		0 %
the first k groups have been assigned people		0 %
the first k groups have been assigned people without anyone being in a group with someone they don't want to work with	1 respondent	20 %