Problem One: RECURSIVE META ALGORITHM 1:

Name of Problem: Maximum Value Contiguous Subsequence.
Write: Problem statement: Given a sequence of n real numbers A(1) ... A(n), determine a contiguous subsequence A(i) ... A(j) for which the sum of elements in the subsequence is maximized.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Maximum Value Contiguous Subsequence

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Two: RECURSIVE META ALGORITHM 1:

Name of Problem: Making Change.
Write: Problem statement: You are given n types of coin denominations of values $v(1) < v(2) < ... < v(n)$ (all integers). Assume $v(1) = 1$, so you can always make change for any amount of money C. Give an algorithm which makes change for an amount of money C with as few coins as possible.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Making Change

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Three: RECURSIVE META ALGORITHM 1:

Name of Problem: Longest Increasing Subsequence.
Write: Problem statement: Given a sequence of n real numbers A(1) ... A(n), determine a subsequence (not necessarily contiguous) of maximum length in which the values in the subsequence form a strictly increasing sequence.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Longest Increasing Subsequence

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Four: RECURSIVE META ALGORITHM 1:

Name of Problem: Box Stacking.
Write: Problem statement: You are given a set of n types of rectangular 3-D boxes, where the i^th box has height h(i), width w(i) and depth d(i) (all real numbers). You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box. Of course, you can rotate a box so that any side functions as its base. It is also allowable to use multiple instances of the same type of box.
Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Box Stacking

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Five: RECURSIVE META ALGORITHM 1:

Name of Problem: Building Bridges
Write: Problem statement: Consider a 2-D map with a horizontal river passing through its center. There are n cities on the southern bank with x-coordinates a(1) ... a(n) and n cities on the northern bank with x-coordinates b(1) ... b(n). You want to connect as many north-south pairs of cities as possible with bridges such that no two bridges cross. When connecting cities, you can only connect city i on the northern bank to city i on the southern bank.
Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Building Bridges

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Seven: RECURSIVE META ALGORITHM 1:

Name of Problem: Multiple Knapsacks
Write: Problem statement: Given n objects each of size s[i] 1<=i<=n (s[i] is an integer) and two knapsacks of size k0 and k1. Determine if there exists a subset of objects that exactly fit into both knapsacks.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Multiple Knapsacks

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Eight: RECURSIVE META ALGORITHM 1:

Name of Problem: Max Value Knapsack
Write: Problem statement: Given n objects each of size s[i] 1<=i<=n (s[i] is an integer), a dollar value assigned to each object v[i], and a knapsack of size k. Determine the maximum cumulative value of a subset of objects that can fit into k. There is no need to fill the knapsack exactly.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Max Value Knapsack

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Six: RECURSIVE META ALGORITHM 1:

Name of Problem: Duplicate Knapsack
Write: Problem statement: Given n objects each of size s[i] 1<=i<=n (s[i] is an integer) and a knapsack of size k. Determine if there exists a subset of objects that exactly fit into the knapsack. An object can be used multiple times.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Duplicate Knapsack

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Nine: RECURSIVE META ALGORITHM 1:

Name of Problem: Balanced Partition
Write: Problem statement: You have a set of n integers each in the range 0 ... K. Partition these integers into two subsets such that you minimize |S1 - S2|, where S1 and S2 denote the sums of the elements in each of the two subsets.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Balanced Partition

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Ten: RECURSIVE META ALGORITHM 1:

Name of Problem: Edit Distance.
Write: Problem statement: Given two text strings A of length n and B of length m, you want to transform A into B with a minimum number of operations of the following types: delete a character from A, insert a character into A, or change some character in A into a new character. The minimal number of such operations required to transform A into B is called the edit distance between A and B.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Edit Distance

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Eleven: RECURSIVE META ALGORITHM 1:

Name of Problem: Counting Boolean Parenthesizations
Write: Problem statement: You are given a boolean expression consisting of a string of the symbols 'true', 'false', 'and', 'or', and 'xor'. Count the number of ways to parenthesize the expression such that it will evaluate to true. For example, there are 2 ways to parenthesize 'true and false xor true' such that it evaluates to true.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Counting Boolean Parenthesizations

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Twelve: RECURSIVE META ALGORITHM 1:

Name of Problem: Optimal Strategy for a Game
Write: Problem statement: Consider a row of n coins of values v(1) ... v(n), where n is even. We play a game against an opponent by alternating turns. In each turn, a player selects either the first or last coin from the row, removes it from the row permanently, and receives the value of the coin. Determine the maximum possible amount of money we can definitely win if we move first.

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Optimal Strategy for a Game

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Fourteen: RECURSIVE META ALGORITHM 1:

Name of Problem: Machine Jobs
Write: Problem statement: I have a set of N jobs, and two machines A and B. Job i takes A[i] time on machine A, and B[i] time on machine B. What is the minimum amount of time I need to finish all the jobs?

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Machine Jobs

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Thirteen: RECURSIVE META ALGORITHM 1:

Name of Problem: Largest contiguous sum
Write: Problem statement: Given an array, find the largest sum along a contiguous segment of it

Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: Largest contiguous sum

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction:

Problem Fifteen: RECURSIVE META ALGORITHM 1:

Name of Problem: DNA alignment
Write: Problem statement: Given two strings A length n and B length m, find the maximally scoring alignment. The score is defined as: insert a gap -2, delete -2, match +1, mismatch A<->G -3, C<->T -3, A<->C -1, G<->T -1
Write: What are the variables that describe all problems?

Write: What is the type of the solution?

Math: Write the function declaration:

Think: What are the simplest problems? What are their solutions?

Math: define the base cases:

Think: How can a larger problem be broken into smaller problems? (Think simply, but creatively)

Math: Write down the simpler problems in terms of the indexes

Think: How is the solution to the input problem constructed from the solutions to the smaller problems?

Math: Write down the solution construction step

GO TO NEXT PAGE

RECURSIVE META ALGORITHM 2:

Name of Problem: DNA alignment

Finish Code: Put it all together

1) Function declaration:
2) Base cases:
3) Combine problem decomposition with solution construction: