Asian MathSci League, Inc (AMSLI)
Partner: National Olympiad in Informatics Philippines (NOI.PH)
Email address: amsliphil@yahoo.com / ask@noi.ph
Contact Nos: +632-9254526 +63906-1186067

Student Copy
AIEP Scratch 2 Session 8

GREEDY ALGORITHM

"Programming isn't about what you know; it's about what you can figure out."
~ Chris Pine

Computer science is one peculiar discipline. On one hand, we have hyper-technical terms, like ***reduced instruction set computer*** and ***peripheral computer interconnect express***. On the other hand, we also have coinages borrowed from Greek mythology, like ***trojan*** (from the Trojan horse), and from political theory, like ***divide and conquer***.

Interestingly, there is a certain class of algorithms known as GREEDY ALGORITHMS, aptly named as such since they satisfy the GREEDY CHOICE PROPERTY:

"The best possible outcome for a given task can be achieved by greedily choosing the best possible outcome for each mini-task."

Suppose that Tiffany is participating in a quiz show where there are five questions worth 30, 20, 10, 40, and 50 points each. Since she prepared and studied hard for the event, she believes that all of the questions are of equal difficulty.

If she is only allowed to answer at most three items, which questions should she choose to answer in order to garner the most number of points?

http://clipart-library.com/clipart/1142235.htm

The answer is, indeed, intuitive, even without any coding background. But, in the interest of a more formal and structured way of thinking, let us dissect the thought process:

a. **What is the mini-task here?** Choosing a question to answer.
b. **How do we greedily choose the best possible outcome for each mini-task?** Choose the unanswered question with the highest number of points.
c. **So what is the answer?** Choose the following questions: 50, 40, and 30 — for a total of 120 points.

Asian MathSci League, Inc (AMSLI)
Partner: National Olympiad in Informatics Philippines (NOI.PH)
Email address: amsliphil@yahoo.com / ask@noi.ph
Contact Nos: +632-9254526 +63906-1186067

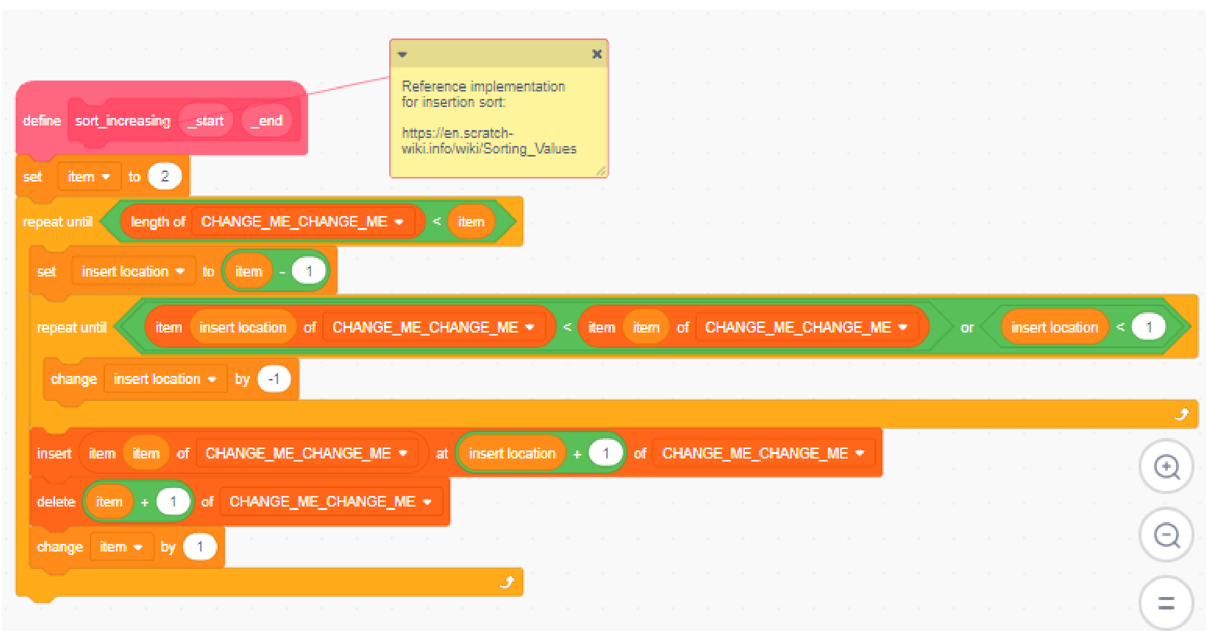
In translating this idea into code, we have to first SORT the points per question and store these sorted values inside a list. **This is a required first step in almost all greedy algorithm questions.** But you do not have to worry; in all the skeleton codes, I have already provided My Blocks for sorting[1] ☺

STEP 1: Select which sorting My Block to use

a) `sort_increasing` for arranging values from least to greatest
b) `sort_decreasing` for arranging values from greatest to least

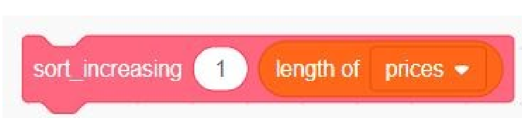
STEP 2: Edit the definition of the My Block

Change all the instances of `CHANGE_ME_CHANGE_ME` to the list that you want to sort.



STEP 3: Call the My Block

The two arguments are 1 and the length of the list to be sorted since our goal is to sort the entire list.



[1] It uses insertion sort, an $O(n^2)$ algorithm that is good if there are only few values to be sorted.

Asian MathSci League, Inc (AMSLI)
Partner: National Olympiad in Informatics Philippines (NOI.PH)
Email address: amsliphil@yahoo.com / ask@noi.ph
Contact Nos: +632-9254526 +63906-1186067

ACTIVITY: Mark and Toys

Taken from Hackerrank Online Judge *(slightly reworded)*

https://www.pinterest.co.kr/pin/480337116513002412/

Mark and Jane are very happy after having their first child. Their son loves toys, so Mark wants to buy some. There are a number of different toys lying in front of him, tagged with their prices. Mark has only a certain amount to spend, and he wants to maximize the number of toys he buys with this money.

If each toy can be bought only once, determine the maximum number of gifts he can buy.

Your sprite should ask the following questions:

a. What is Mark's budget?
b. How many toys are in the shop?
c. Enter toy price *(this will be repeated depending on the number of toys)*.

Suppose Mark's budget is 50 and there are 7 toys with the following proces: 1, 12, 5, 111, 200, 1000, and 10. Mark should then buy the toys worth 1, 12, 5, and 10 — for a total of 4 gifts. The expected output is shown below:

Prepared by ***Mark Edward M. Gonzales***

ACTIVITY: Minimum Array Difference

Taken from Hackerrank Online Judge *(reworded)*

Given a list of numbers, find the minimum difference between any two numbers in this list. The difference is always taken to be the absolute value.

Your sprite should ask the following questions:

a. How many elements will your array contain?
b. Input a number*(this will be repeated depending on the number of elements)*.

Suppose the list (array) will contain 5 values and these numbers are 100, 50, 63, 104, and 34 The minimum difference between any two numbers is between 100 and 104 — which is 4. The expected output is shown below:

ACTIVITY: Shopaholic

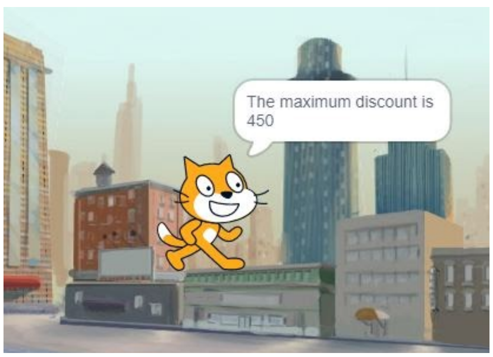
Taken from the 2007 Nordic Collegiate Programming Contest *(slightly reworded)*

Lindsay is a shopaholic. Whenever there is a discount of the kind where you can buy three items and only pay for two, she feels a need to buy all items in the store. You have realized that the stores with discount offers are quite selective; it is always the cheapest items that you can get for free.

http://clipart-library.com/shopping-cliparts.html

As an example, when your friend comes to the counter with seven items, costing 400, 350, 300, 250, 200, 150, and 100 dollars, she will have to pay 1500 dollars. In this case she got a discount of 100 + 150 = 250 dollars — the two cheapest items have been deducted.

You realize that if she goes to the counter three times, she might get a bigger discount. For example, if she goes with the items that costs 400, 300 and 250, she will get a discount of 250 the first round. The next round she brings the item that costs 150 giving no extra discount, but the third round she takes the last items that costs 350, 200 and 100 giving a discount of an additional 100 dollars, adding up to a total discount of 350.



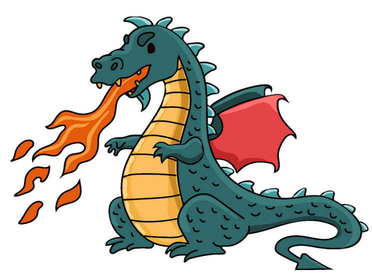
Your sprite should ask the following questions:

a. How many items is Lindsay going to buy?
b. Enter item price *(this will be repeated depending on the number of items)*.

Your job is to find the maximum discount Lindsay can get. In the seven-item scenario presented above, she can have a discount of at most 450. *How?*

☆ BOSS-LEVEL CHALLENGE: Dragon of Loowater ☆

Taken from the 2007 Waterloo Programming Contest[2]



Once upon a time, in the Kingdom of Loowater, a minor nuisance turned into a major problem.

The shores of Rellau Creek in central Loowater had always been a prime breeding ground for geese. Due to the lack of predators, the geese population was out of control. The people of Loowater mostly kept clear of the geese. Occasionally, a goose would attack one of the people, and perhaps bite off a finger or two, but in general, the people tolerated the geese as a minor nuisance.

[2] Image from https://creazilla.com/nodes/14981-dragon-clipart.
The original problem can be accessed from https://open.kattis.com/problems/loowater. Input specfications have been simplified for pedagogical purposes.

One day, a freak mutation occurred, and one of the geese spawned a multi-headed fire-breathing dragon. When the dragon grew up, he threatened to burn the Kingdom of Loowater to a crisp. Loowater had a major problem. The king was alarmed, and called on his knights to slay the dragon and save the kingdom.

The knights explained: "To slay the dragon, we must chop off all its heads. Each knight can chop off one of the dragon's heads. The heads of the dragon are of different sizes. In order to chop off a head, a knight must be at least as tall as the diameter of the head. The knights' union demands that for chopping off a head, a knight must be paid a wage equal to one gold coin for each centimetre of the knight's height."

Would there be enough knights to defeat the dragon? The king called on his advisors to help him decide how many and which knights to hire. After having lost a lot of money building Mir Park, the king wanted to minimize the expense of slaying the dragon. As one of the advisors, your job was to help the king. You took it very seriously: if you failed, you and the whole kingdom would be burnt to a crisp!

Your sprite should ask the following questions:

a. How many heads does the dragon have?
b. Input the diameter of a head *(repeated depending on the number of heads).*
c. How many knights does Loowater have?
d. Input the heights of a knight *(repeated depending on the number of knights).*

Say the minimum number of gold coins that the king needs to pay to slay the dragon. If it is not possible for the knights to slay the dragon, say `"Loowater is doomed!"`

If there are 2 dragon heads (diameters: 5 and 4) and 3 knights (heights: 7, 8, and 4), then the minimum number of gold coins is 11. However, if there are 2 dragon heads (diameters: 5 and 5) and 1 knight (10), then Loowater is doomed!

Asian MathSci League, Inc (AMSLI)
Partner: National Olympiad in Informatics Philippines (NOI.PH)
Email address: amsliphil@yahoo.com / ask@noi.ph
Contact Nos: +632-9254526 +63906-1186067

#Unplugged: Not Your Ordinary Code

Can you decode the following message written in Morse code: • – – – • • • • • • • ? It can either be ABS (• – then – • • • then • • •) or EGEES (• then – – • then • then • then • • •). Oh no! 😰

This problem is resolved by using so-called PREFIX CODES. Once a prefix (a set of starting letters) appears in a code, it cannot be used again, thus eliminating the ambiguity that we encountered earlier. For example, Morse code is NOT a prefix code since the code for F (• • – •) contains the prefix • •. But • • is also the code for the letter I.

What then is an example of a prefix code? For example, we have a simplified alphabet with only three letters A, B, and C. If A is • – •, B is • – –, and C is – • – – –, then there is no room for confusion, and we have a prefix code.

Now, there are some things to consider when creating a good prefix code, such as:

a. Letters that are used frequently should have shorter codes.
b. We want our codewords to be as short as possible.

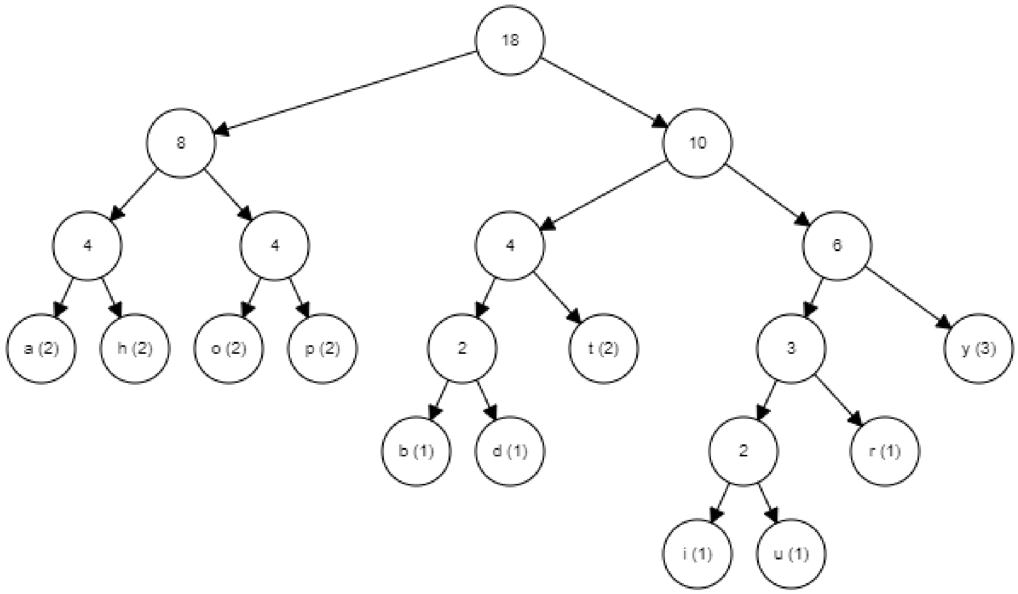
The quest for making good prefix codes started in the last century — and it was not easy. In the 1950s, a professor at Massachusetts Institute of Technology, Robert M. Fano, gave his class two options: take a final exam or write a paper on the said problem. His student, David A. Huffman, opted for the latter but struggled to construct an efficient prefix code.

But, on the verge of giving up, he had a eureka moment! In 1952, he published his findings and succeeded in something that his teacher Fano and Claude E. Shannon (the Father of Information Theory) failed to solve. Until now, his idea is used all around the world for data compression. The algorithm now bears his name: the HUFFMAN CODING SCHEME.

READING THE HUFFMAN TREE

Today, we will learn about the Huffman coding scheme. Go to the following website, and try entering `happybirthdaytoyou`: https://people.ok.ubc.ca/ylucet/DS/Huffman.html

Try following the steps in the animation. The idea here is that, the two letters with the least frequencies are fused into a tree. These two frequencies are added, and the sum becomes the root node of the said tree. This process continues until only a single tree is left.



Reading the Huffman tree

The figure above shows the final Huffman tree generated using `happybirthdaytoyou` as our word. In order to get the code for each letter, travel from the root node all the way up to the leaf node representing the letter — in other words, we trace the steps from the top of the tree to our goal letter. Going to the left is a 0 while going to the right is a 1.

Let us take a look at some examples:

a. To reach letter a, we travel left-left-left. Hence, the code for a is 000.
b. To reach letter d, we travel right-left-left-right. Hence, the code for d is 1001.
c. To reach letter u, we travel right-right-left-left-right. Hence, the code for u is 11001.

Therefore, if we want to encrypt the word "dua," our codeword is 100111001000.

Challenge

Build a Huffman tree for the word `amsliinformaticsenrichmenttrainingprogram`.

- What is the codeword for `aiep`?
- Why is Huffman coding considered a greedy algorithm?
- Observe that, in every step, we are only concerned with the two least frequencies. Does it make sense to sort all the values after every step?