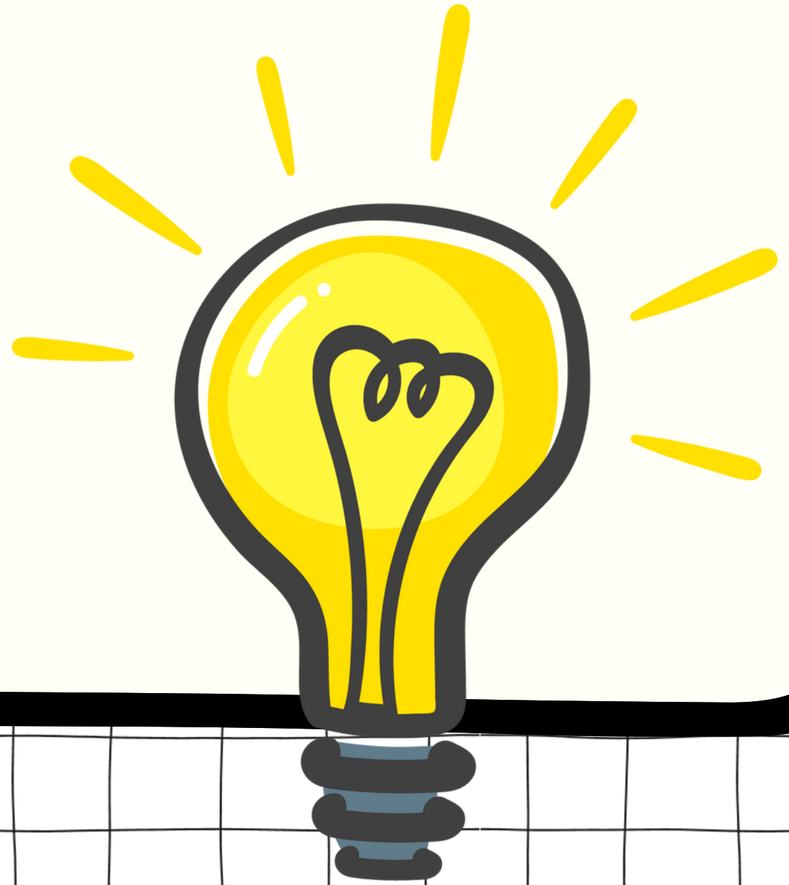
Case Study 1

Borja, Cu, Marquez K.



Determining how many days they can procastinate/delay completing an assignment while still finishing it on time.

Problem

If you are given an assignment, how can you find the most number of days you can procrastinate before a deadline given that there is a constant rate of progress?

Decomposition

- Interface for User.
- Creating and Implementing Algorithm via Dynamic Programming.

Pattern Recognition

- As the deadline draws nearer, the maximum time to procrastinate decreases.

Abstraction

- Relevant Information: deadline, rate of progress, progress requirement.
- Less Relevant: name and nature of assignment.

Problem

How can the procrastination problem be implemented using dynamic programming techniques?

Decomposition

- Implement a bottom up approach.
- Create a list that will store the progress per day.
- The length of the list will be determined by the deadline.
- Start by calculating & storing the progress at day 1.

Pattern Recognition

- The addition of progress per day is repetitive and it can be compared to the required progress.

Abstraction

- Relevant Information: available programming techniques.
- Less Relevant: programming language.

1. This function getInput() which is used to collect input from the user regarding their assignment details. It prompts the user to input the total number of pages in the assignment, the rate at which they can complete pages per day, and the number of days until the assignment deadline.

```
[ ]  def getInput():
         # Function to get inputs from the user
         pages = int(input("How many pages is your assignment? ")) # Input: Total number of pages in the assignment
         rate = int(input("How many pages can you do per day? "))   # Input: Rate of pages completed per day
         deadline = int(input("In how many days is the deadline? ")) # Input: Deadline for the assignment
         return pages, rate, deadline
```

2. The function compute that calculates how many days one can procrastinate on a task given the total number of pages to be completed, the rate of completion, and the deadline. It uses a loop to simulate daily progress and returns the number of remaining days until the deadline if the task can be completed, or -1 if it cannot.

```python
def compute(pages, rate, deadline):
    table = [0] * (deadline + 1) # Initialize a list to store progress per day
    for day in range(1, deadline + 1): # Iterate over each day until the deadline
        table[day] = table[day-1] + rate # Calculate the total progress up to the current day
        if table[day] >= pages: # If the total pages are reached or exceeded
            return deadline - day # Return the remaining days left until the deadline
    print("You cannot procrastinate") # If the assignment cannot be completed by the deadline
    return -1 # Return -1 indicating inability to procrastinate

# Main part of the code
inputA, inputB, inputC = getInput() # Get inputs from the user
print("You can procrastinate for " + str(compute(inputA, inputB, inputC)) + " days") # Print the result
```

```
How many pages is your assignment? 20
How many pages can you do per day? 5
In how many days is the deadline? 6
You can procrastinate for 2 days
```