# Monte Carlo Simulation

## Introduction

This document provides a brief and clear explanation of two Monte Carlo simulation functions and for the created tool : one for throughput/leadtime and the other for story points.

## Throughput/Leadtime Simulation

Throughput represents the number of items completed in each period, and leadtime represents the time taken to complete an item. Here's how the function works:

1. Throughput and leadtime data are converted to numpy arrays for efficient handling.

2. Random samples are taken from the throughput and leadtime data, and then adjusted by a focus factor to simulate the completion of items in the backlog.

3. The adjusted throughput is added to the number of items completed, and the leadtime is added to the time elapsed for each iteration.

4. The simulation continues until the backlog is completed, and the results (time elapsed when the backlog is completed) are returned.

Overall, in the ***monte\_carlo\_simulation*** function, the random sampling from these datasets helps in simulating different scenarios for the backlog completion, based on historical performance data. Throughput helps in understanding how many items are being completed in a given period, while leadtime helps in understanding how long it takes to complete those items. Together, they offer a comprehensive view of the process, enabling more accurate forecasts for project completion.

## Story Point Simulation

This function is similar to the throughput/leadtime-based simulation but adapted for story points. Here's how it works:

1. Random samples are taken from the story point series, dropping NaN values, and adjusted by the focus factor.

2. The adjusted story points are added to the total story points until the backlog items are reached.

3. The results (total story points) are returned.

# Forecast Table Calculation

The `calculate\_forecast\_table` function takes the results of a Monte Carlo simulation and a start date as input and calculates the forecast table. It defines probability intervals and calculates the elapsed time for each interval based on the simulation results. It then calculates the corresponding completion dates, cost of delay, and organizes these values into a DataFrame. The probability intervals, elapsed times, completion dates, and cost of delay are all reversed for descending order in the resulting DataFrame. This gives a complete view of the probable outcomes, allowing for better decision-making.