# Missouri State University
## Department of Computer Science

Assignment 3: Evolutionary Computing
Summer 2025

## Submission Instructions

Students must follow these submission guidelines to receive full credit:

1. **Submission:**
   - Submit a single compressed '.zip' file containing all source files (Python scripts) and PDF report for conceptual questions.
2. **File Naming Convention:** Name your '.zip' file as: `FirstName_LastName_EC_AS3.zip`
3. **Deadline:** Submit your assignment by July $5^{th}$ 2025.
4. **Academic Integrity:**
   - Your submission must be your own work.
   - Plagiarism will result in a zero grade and possible disciplinary action.

## Objective

SwiftShip, an e-commerce company, processes hundreds of customer orders daily. Each order consists of items with varying weights, and each shipping box has a maximum weight capacity of 10 kg and can carry multiple items (orders). Due to rising packaging and courier costs, SwiftShip wants to minimize the total number of boxes used, while ensuring that no box exceeds its weight limit.

You have been tasked with designing an optimization system using a Genetic Algorithm to automatically assign items to boxes in a way that:

- Minimizes the number of boxes used, and
- Satisfies the weight constraint (i.e., total weight per box $\leq$ 10 kg).

Your algorithm should handle various order sizes, such as 10, 25, 50, and 100. Make sure to initialize weight for each order between 0 to 2 kg.

## Assignment Tasks

### Q1. Encoding and Initialization

- Each solution (individual) is represented by a list of integers indicating bin assignments. For example: `[0, 1, 0, 2, 1, 1, 2, 1, 3, 0]`.
- This means that 10 orders are assigned to bins 0 through 3. (Orders can be 10, 25, 50, and 100).
- You can assume the maximum number of boxes (bins) equals the number of items.
- Implement a function to generate a random initial population of such individuals.

## Q2. Function Evaluation

1. Implement a function to evaluate the objective function $f$, defined as the number of unique bins used (i.e., the goal is to minimize $f$).
2. Also compute the constraint violation $g$, defined as the number of bins whose total weight exceeds 10 kg. Examples:
   - If no bin exceeds the limit, $g = 0$ (feasible).
   - If one bin exceeds the limit, $g = 1$.
   - If all bins exceed the limit, $g = N$ (worst case).
3. Each individual will thus be evaluated using both $f$ and $g$.

## Q3. GA Operations

Implement the following Genetic Algorithm components:

1. **Tournament Selection:**
   - For each selection, choose two individuals at random and determine a winner.
   - Repeat until the mating pool has $N$ individuals.
   - Then select pairs from the mating pool for crossover.
2. **Winner Selection Rules:**
   - If both individuals are feasible ($g_1 = 0$, $g_2 = 0$), the one with lower $f$ wins.
   - If one individual is feasible and the other is not, the feasible one wins.
   - If both are infeasible, the individual with the lower $g$ wins.
3. **Crossover:** Implement one-point or two-point crossover between two selected parents.
4. **Mutation:**
   - For each gene in the chromosome, mutate it with a probability $p_m = \frac{1}{\text{number of boxes}}$.
   - Mutation means reassigning the gene to a new random bin (between 0 and $N - 1$).
   - Ensure that genes only hold integer values.

## Q4. GA Execution

- Run the GA for 50 generations with a population size of 20 considering 10, 25, 50, and 100 orders separately.
- Vary the number of population size and max generation to see the effect on performance.
- Plot the best and average fitness (i.e., number of bins used) per generation.
- Report the best solution found and its bin-wise packing configuration.

## Q5. Analysis and Comparison

1. Compare the convergence behavior of different runs (e.g., different random seeds or order sizes).
2. Discuss whether the algorithm tends to find feasible solutions early or late in the process.