# Objective

The risk-based test strategy suggests splitting overall work on a new feature into items, measure risk and test handling time associated with each item, and select resulting test coverage in such a way that the highest possible volume of risk is covered by test in budget of time available to test org for testing of that new feature. Depending on implementation of the risk-based testing approach and depending on a project stage on which it is involved, new features, feature functional requirements, or test cases can be used as items. Risk estimates associated with each item, and estimation of time required to cover that item, are out of scope of this work.

Our objective is to build a software tool that automates process of selection of items for execution based on existing risk factor and time budget associated with each item.

*Overall objective of a tool at a high level can be described by the knapsack problem.*

*The knapsack problem or rucksack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible. It derives its name from the problem faced by someone who is constrained by a fixed-size knapsack and must fill it with the most valuable items. (see Wikipedia).*

*In our case, “weight” is an estimate of execution time, “value” is a risk level associated with each item, and “given limit” is an overall time budget available for handling set of items.*

RBTCS tool provides the ability to input data as an excel file, and generates output as an excel file containing list of items to be handled by test.

# High-level Functional Requirements

|  |  |  |
| --- | --- | --- |
| **ID** | **Description** | **Status** |
| 1 | Input data as an excel file containing list of items and associated risk factor and execution time | Approved |
| 2 | Output data as an excel file containing list of selected items | Approved |
| 3 | Support command line arguments (filename, names of columns, available time budget) | Approved |
| 4 | Implement optimal algorithm for item selection: use dynamic programing approach for 0/1 knapsack problem:   * O(NW) for time complexity; * O(NW) for space complexity * N – is the number of items; * W – is the available time budget; | Approved |
| 5 | Implement sub-optimal algorithm if complexity for optimal is too high: use greedy algorithm:   * O(NlogN) time complexity; * O(N) space complexity; | Approved |
| 6 | Implement logging | Approved |
|  |  |  |

# Use Cases

The main use case for the tool is as follows:

1. User prepares input file in excel. Input file contains columns with estimated risk-factor and execution time for each item.
2. User run the tool, and in command line arguments specifies input file name, column names containing risk factor, execution time and column name where to put selection marks. User also provides overall time budget.
3. Tool produces output excel file which contains same data but with selection column filled (i.e. with data that shows which items have been selected for coverage, and which items haven’t been selected. Tool also shows risk coverage.

# Functional Specification

## Input data file specification

Input data file can either be .xlsx or .xls format.

Input data file contains a table:

* First row of the table contains names of columns;
* Subsequent rows contain items and information associated with them;
* At minimum 4 columns are required: ID, risk-factor, execution-time, selection.
  + ID uniquely identifies items;
  + Risk-factor contains estimate of risk level associated with each item;
  + Execution time contains estimate of execution time for each item;
  + Selection column contains output results of optimization algorithm and shows which items were selected, and which not.