Task 1 – Layout Application, Command Line Arguments (2 points)
The Gene Ontology (GO) is a curated vocabulary to describe gene functions. Its entries are stored in .obo files (plain or gzipped). Some terms are deprecated and marked with is_obsolete: true. Often, alternatives are suggested using consider.Create the basic layout of a command-line application with main function, help function and checking of command-line arguments. Save the possible command line arguments in variables or use a command-line processor like an argparser. Check the number and content of arguments. If the number of arguments was not given call the help function and exit the application.The application must accept an option name (e.g.,--consider-table or-- obsolete-stats, hierarchical_relationship), an input filename ending in .obo or .obo.gz.
An optional namespace molecular_function, cellular_component, biological_process. Validate that the input file exists, check that the given option is allowed, and ensure that an optional argument is a valid namespace. If input is invalid, print a help message and terminate.

Task 2 – Consider Table (8 points)
Our program should work with any GOobo file. Please don't hardcode the filename in your application. Implement a considerTable function that returns a tabular output for all obsolete GO terms and their offered alternative GO id(s) given on consider lines in the GO obofile. If there are no alternative GO's, give NA's.
Implement a replacedby function which returns a tabular output for all obsolete GO terms and their offered alternative GO id(s) which are given on replaced by line in the GO obofile. If there are no alternative GO's, give NA's. If the user provides, for example, the three arguments: --replaced-by or --consider-table filename.obo molecular_function on the command line, then the GO-obo file is parsed, and only information for GO-ids belonging to this namespace should be displayed to the terminal. The function should itself not print the data to the terminal, but it should return the results, for instance, as a nested vector. Hence, if your code is very slow, limit your search to the first 100 entries to save your programming time during development. You should uncomment this in your final program.

Task 3 - Advanced functionality (8 points)
Extend your application as a C++ class. Write a subset function that returns the number of subsets for each subset in a given filename. Write a xrefsearch function that filters GO-ids and EC-ids based on the Reactome database. Implement the obsoleteStats functionality, which should show the number of obsolete GO entries and how many of them have alternative consider id(s), have been replaced by id(s), and how many have no entries (NA). Implement the hierarchical_relationship function that returns GO_ids with is_an and part_of relationships.