

**Course Title in Hebrew:** מולטי-תכנות לעיבוד מקליבלי

**Course Title in English:** Multiprogramming for Parallel Processing

**Course Code:** 202-1-5401

**Credit Hours:** 4.0

**Course Instructor:** Dr. Ran Atiniger

**Pre-requisites:** 202-1-1-20

**Course Description:**

The course presents a method for planning and developing algorithms and computer programs for parallel processing directly from a specification (specification). In this way, the correctness of the program (that is, whether it satisfies the properties stated in the specification) follows directly from the steps in the program construction.

The process of constructing parallel programs (multi-programs) taught in this course is based on a simple theory in which we know what we need to prove the correctness of each step. If we perform according to this method in each step of the development, we will get a multi-program that is proven correct. The main theory (specification) is the idea of interference freedom (interference freeness) between individual processes. According to this idea, if two processes communicate through a shared memory, the correctness of one process is not affected by the other process. If we can prove that the code of the other process does not affect the correctness of the first process, the proof obtained in this way ensures partial correctness: when the multi-program stops, all variables will satisfy the properties stated in the specification. In addition, we will learn development techniques to avoid difficult situations of interference (deadlock) and restart variables in shared memory and others.

The course is based entirely on the book "On a Method of Multiprogramming" by G. Feijen and van Gasteren. In addition to learning the development method, the course (like the book) demonstrates the method by developing a wide range of parallel algorithms, while studying parallel programming techniques to solve synchronized problems, interference, and deadlock.

"Feijen and van Gasteren" "On a Method of Multiprogramming".
The following basic topics will be covered:

- A Computational Model
- Program Notation and its Semantics: Hoare triples and weakest-liberal-preconditions
- The Core of the Owicki/Gries Theory: annotating a multiprogram; postconditions
- Private Variables and Orthogonality
- System Invariants
- Mutual Exclusion
- Co-assertions and Strengthening the Annotation
- Synchronization and Total Deadlock
- Individual Progress and the Multibound

As time and progress permit, a modest selection of algorithms will be studied (both in class and home assignments), taken from the following collection (as it appears in the second part of the course textbook, starting at Chapter 13):

- The Safe Sluice
- Peterson’s Two-Component Mutual Exclusion Algorithm
- Handshake Protocols
- Phase Synchronization for Two Machines
- The Parallel Linear Search
- The Initialization Protocol
- The Non-Blocking Write Protocol
- Mutual Inclusion and Synchronous Communication
- A Simple Election Algorithm
- Peterson’s General Mutual Exclusion Algorithm
- Monitored Phase Synchronization
- Distributed Liberal Phase Synchronization
- Distributed Computation of a Spanning Tree
- Shmuel Safra’s Termination Detection Algorithm
- The Alternating Bit Protocol

The assignment, midterm examination, and final project are all theoretical, in the sense that any amount of programming will be done “on paper”. The assignment and project are to be prepared in (small) teams.

The final grade will be determined by one homework assignment (20%), a must-pass midterm examination (20%), and a final project (60%).

The course is exclusively based on the monograph *On a Method of Multiprogramming*, by W.H.J. Feijen and A.J.M. van Gasteren.