Sample Ada Programs

Some of these will be covered in class.

A simple "hello world" program in a single procedure with a trivial Ada task. To compile and run, type "gnatmake hello" and then "./hello".

- hello.adb

A simple single-procedure program with a task and the procedure body running concurrently. To compile and run, type "gnatmake f" and then "./f".

- f.adb

A version of the above simple program in which the task has an entry. To compile and run, type "gnatmake f2" and then "./f2".

- f2.adb

A simple program that has:

- a stack package (package spec and body) defining push and pop, and
- a main procedure that calls push and pop.

To compile and run, type "gnatmake main" and then "./main" (you'll need to enter two integers, both less than 20, which determines how many items are pushed on the stack and how many are popped, respectively).

- stack.ads
- stack.adb
- main.adb

A version of the above stack program, except the stack type is an abstract data type (i.e. a private type). Thus, multiple stack variables can be declared. It contains:

- an adt_stack package (package spec and body) which declares a stack type and implements push and pop for that type.
- an adt_main procedure that declares several stack variables and calls push and pop on them.

To compile and run, type "gnatmake main" and then "./main" (you'll need to enter two integers, both less than 20, which determines how many items are pushed on the stack and how many are popped, respectively).

- adt_stack.ads
- adt_stack.adb
- adt_main.adb

A trivial semaphore program that has:

- a binary semaphore package (package spec and body) with procedures P and V, which uses a simple task to enforce the correct behavior. The semaphore itself is a private type and the task contains a select/terminate construct so that the program ends when there is nobody left to call P or V.
- a procedure testsem that calls P and V.

To compile and run, type "gnatmake testsem" and then "./testsem".

- sem.ads
- sem.adb
- testsem.adb

A modified stack program that handles push and pop in a multi-tasking environment. It contains:

- a new_stack package defining push and pop, which uses a task to enforce mutual exclusion and
- a test_new_stack procedure that spawns multiple tasks, each calling push and pop.

To compile and run, type "gnatmake test_new_stack" and "./test_new_stack".

- new_stack.ads
- new_stack.adb
- test_new_stack.adb

Another modified stack program that handles push and pop in a multi-tasking environment. This one uses protected types to enforce mutual exclusion (specifically, protected procedures push and pop). It contains:

- a protected_stack package defining protected procedures push and pop.
- a protected_test procedure that spawns multiple tasks, each calling push and pop.

To compile and run, type "gnatmake protected_test" and "./protected_test".

- protected_stack.ads
- protected_stack.adb
- protected_test.adb

A program illustrating the use of task types.

- test_task_types.adb

To compile and run, type "gnatmake test_task_types" and "./test_task_types".