CS 201 Section 1 Lab 1 – due 6pm, Monday, January 24, 2000

One of the (few) features of machine code that is not readily accessible from higher-level languages (HLLs) is testing for overflow on integer arithmetic operations. Most HLLs just quietly allow overflow, and leave it to the user to discover the problem by inspecting the output (maybe!).

Your task is to write a short, simple assembly language subroutine, suitable for calling from C, that sums an array of integers and returns the result – unless one of the addition operations causes an overflow, in which case an error status flag is returned instead.

Here's the C interface for your routine:

```
int safesum(int size, int a[], int *result);
```

Your routine should add together the array elements a[0],...,a[size-1]. If no overflow occurs, the resulting sum should be stored in result, and the routine should return 1 (i.e., "true"). If overflow occurs, nothing should be stored in result, and the routine should return 0 (i.e., "false.") Note that ordinary C `int`'s correspond to 4 byte signed words in machine language.

A fragment of C code that uses `safesum` might look like this:

```
int a[5] = ...;
...
int sum;
int status = safesum(5,a,&sum);
if (status)
  printf("Result is %d\n", sum);
else
  printf("Overflow\n");
```

You are responsible for writing both `safesum` itself and a a test driver in C that provides *brief* but convincing evidence that your `safesum` works correctly on a small sample of interesting arrays. You may find the file `/usr/include/limits.h` useful for your driver.

Your implementation of `safesum` should be a **leaf** routine, as described in class. You must not use register windows. You should try to write the shortest and most efficient routine you can. In particular, make sure you fill the delay slots with useful instructions.

To submit this lab, you need to send two files: `safesum.s`, containing assembly code for your routine, and `driver.c`, containing your driver code. Bundle these together using `shar` and mail to `cs201-1acc`, e.g.,

```
shar safesum.s driver.c | mail -s "lab1" cs201-1acc
```

Your submission will be built by using the command

```
cc -o safetest driver.c safesum.s
```

and then tested by running `safetest`.