Stanley Adom

Talen Phillips

Lab 10: Arduino Memory Usage Using Recursion

16APR2013

**Introduction**

In this lab, students were asked to create a program to demonstrate memory usage with a function. They were instructed to create a function that calculates a factorial with recursive function calls, and have that function print the ram usage to the serial monitor.

**Procedure**

First, the students were given a function that calculated the remaining memory available in the Arduino's internal RAM. This function was named freeRam, and can be seen in the code.

Next, the students created a recursive function call that returned a factorial number. Within this function was added the freeRam function call, which printed its value to the screen.

Next, the students created an array that took a numeric input of up to 4 digits, and transformed it into a single integer value. Much of this was provided by the lab assistant.

Here is a copy of the code:

int fact;

void setup() {

// initialize serial communication:

Serial.begin(9600);

}

// this is the function provided by the instructor to display the remaining ram

int freeRam () {

extern int \_\_heap\_start, \*\_\_brkval;

int v;

return (int) &v - (\_\_brkval == 0?

(int) &\_\_heap\_start : (int) \_\_brkval);

}

// This is the recursive function that calculates factorials.

// Note the freeRam function call that's printed to the screen.

int fact\_funct (int input){

Serial.println(freeRam ());

if (input>=1){

return fact = input \* fact\_funct(input-1);

}

else return fact;

}

void loop() {

int in\_arr[4]; // input array

int place = 1;

int input = 0;

int i;

fact = 1;

Serial.println("Enter a number: ");

while (Serial.available() != 1);

// this loop feeds the user's digits into the array

for (i=0; Serial.available() > 0; ++i){

in\_arr[i] = Serial.read() - '0';

delay(10);

}

// this loop turns the values in the array into an integer value

for (--i; i>=0; --i){

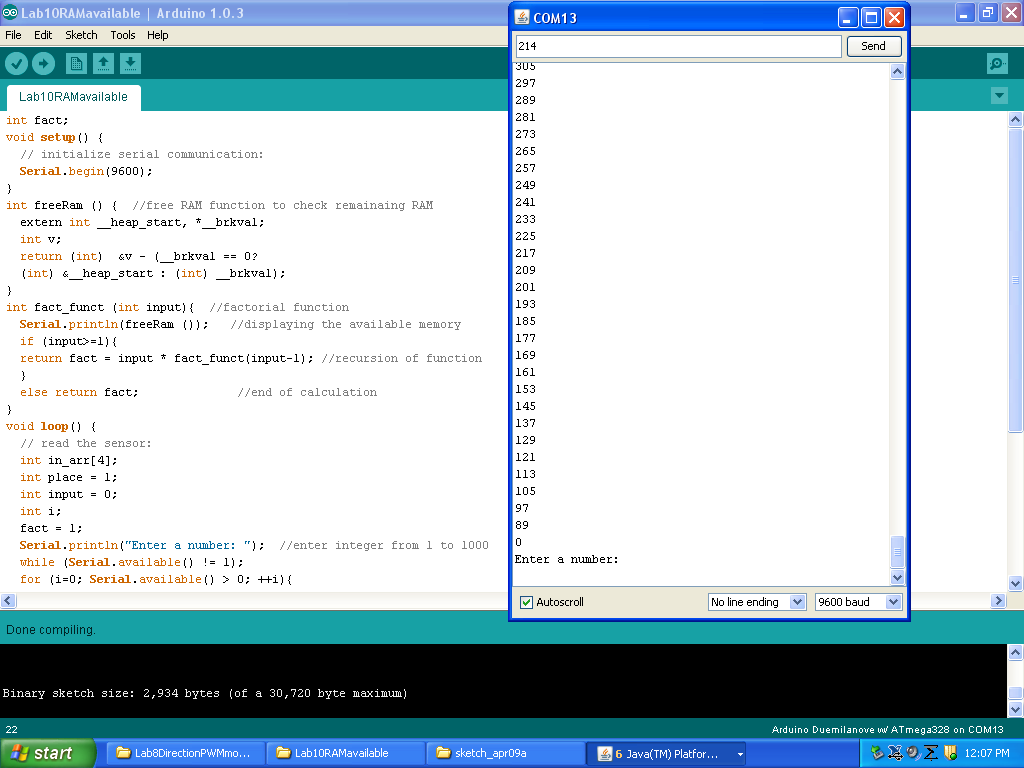
input += in\_arr[i] \* place;

place \*= 10;

}

Serial.println(fact\_funct(input)); //this starts the loop

}

Here is a screenshot of the running code:

**Conclusion**

In this lab, the students learned a more effective method of using the Arduino microcontroller's serial interface to take data from user input. They used the integer inputs to calculate a factorial. As well as the factorial, the number was used to show the effects of overrunning the available memory in the onboard RAM of the arduino. What number caused the arduino to overflow and crash depended on what kind of variable was used. For example, one can display far larger factorials with a double than with an int type, but the double type uses memory at a much faster rate, and thus crashes the program with far fewer iterations. In my program, the int type was used, and it crashed at about 213 iterations. The minimum memory before the program crashed remained constant at 89.