#include <avr/pgmspace.h>

//PROGMEM keyword creates the object in the flash memory instead of RAM

const char array1[] PROGMEM = "Array 1";

// put some strings in RAM initialized data area

char array2[] = "Array 2";

char array5[] = "Array 5"; //this will not be allocated memory if it is not used somewhere

char array6[] = "Array 6";

void setup() {

Serial.begin(9600); // Setup for Serial output

char array3[] = "Array 3"; // put a string in RAM stack area

char \*array4 = (char\*) malloc(strlen("Array 4") + 1);

strcpy(array4, "Array 4"); // put a string in RAM heap area

// Part 1: print address ranges for each memory space

/\*

Serial.print("\nEnd of Flash: ");

Serial.println(FLASHEND, HEX);

Serial.print("End of RAM: ");

Serial.println(RAMEND, HEX);

Serial.print("End of EEPROM: ");

Serial.println(E2END, HEX);

Serial.println();

\*/

// Part 2: Arrays

/\*

Serial.println("String in program memory: ");

char c;

for (int i = 0; (c = pgm\_read\_byte(&array1[i])) != 0; i++) {

Serial.print(c);

}

Serial.println(array1); // accesses RAM - not program memory

Serial.print("Address of string in program memory: ");

Serial.println((int) &array1[0], HEX);

Serial.println();

Serial.println("String in RAM initialized data: ");

Serial.println(array2);

Serial.print("Address of string in RAM initialized data: ");

Serial.println((int) &array2[0], HEX);

Serial.println();

Serial.println("String in RAM stack: ");

Serial.println(array3);

Serial.print("Address of string in RAM stack: ");

Serial.println((int) &array3[0], HEX);

Serial.println();

Serial.println("String in RAM heap: ");

Serial.println(array4);

Serial.print("Address of string in RAM heap: ");

Serial.println((int) &array4[0], HEX);

Serial.println();

//Serial.println(array5);

Serial.println(array6);

\*/

// Part 3: print out the ram

/\* Example function calls:

displayRAM((char \*) 0x100, (char \*) 0x200, false);

//displays memory in 0x100 blocks with 2 second delays

displayAllRAM(2000, false);

\*/

//Part 4(OPTIONAL): What endian?

/\*

unsigned long a = 0x12345678;

unsigned long e = (unsigned long)&a;

Serial.print("Long location in RAM Stack :");

Serial.println(e, HEX);

displayRAM( (char \*)(e - 15), (char \*)(e + 15), true);

displayRAM( (char \*)(e - 15), (char \*)(e + 15), false);

\*/

}

void loop() {} //notice we still have the loop function to keep the compiler happy

void displayAllRAM(int waitTime, bool hex) {

for (char \*i = (char \*) 0x0; i < (char \*) RAMEND; i += 0x100) {

displayRAM(i, i + 0x100, hex);

delay(waitTime);

}

}

/\* example call displayRAM((char \*) 0x8E0, (char \*) 0x8FF, true);

\* if hex is false, characters will be printed, and all other values will be represented as '.' \*/

void displayRAM(char \*start, char \*endd, bool hex) {

char \*array;

for(array = start; array < endd; array += 0x10) {

//create row number

if (array < (char \*)0x10)

Serial.print('0');

if (array < (char \*)0x100)

Serial.print('0');

Serial.print((int)array, HEX);

Serial.print(": ");

//for each index (0 through 15 inclusive)

for(int i = 0; i < 0x10; i++) {

if(hex) {

if (array[i] >= 0x00 && array[i] < 0x10)

Serial.print('0');

Serial.print(array[i] & 0xFF, HEX); //0xFF is our bitmask

} else {

Serial.print((array[i] >= ' ' && array[i] <= 'z') ? array[i] : '.');

}

Serial.write(' ');

}

Serial.println();

}

}