CS 241 Lab 08
(Serial Communication)

Benjamin Stream
Solomon Himelbloom

March 22, 2021

1 Answers to Questions

- **Assignment 0:** Sketch uses 2020 bytes (6%) of program storage space. Maximum is 32256 bytes. Global variables use 346 bytes (16%) of dynamic memory, leaving 1702 bytes for local variables. Maximum is 2048 bytes.
- **Assignment 1:** Sketch uses 6022 bytes (18%) of program storage space. Maximum is 32256 bytes. Global variables use 614 bytes (29%) of dynamic memory, leaving 1434 bytes for local variables. Maximum is 2048 bytes.
- **Assignment 2:** Sketch uses 8002 bytes (24%) of program storage space. Maximum is 32256 bytes. Global variables use 1287 bytes (62%) of dynamic memory, leaving 761 bytes for local variables. Maximum is 2048 bytes.

2 Arduino Commands

- **KEY:** Arduino Task — Your Serial Command Name — Observed Effects

- analogRead on an analog pin. — ”A0” through ”A5” — Reads voltage from 0-1023.
- Turn the pin 13 debug LED on. — “on” — The result is observed as intended: turns the debug LED on.
- Turn the pin 13 debug LED off. — “off” — The result is observed as intended: turns the debug LED off.
- Print ”x” to the serial port 10000 times. — “x” — (What happens while it’s printing?) Prints as you would expect; on the same line vs. new line changes the output to the console without causing an overflow.
- Change the serial baud rate to 19200, and print ”are you reading this?”. — “baud” — Due to the change in baud rate (being written at a different rate than it is being read), the text turns into illegible letters and symbols.
- Test command names with Unicode chars (print the chars in hex). — ”grüning” — (Is this UTF-8?) Prints the respective command with special characters in UTF-8 format. We use unsigned char to remove excessive question marks (FF).
- Use ”new” to allocate 512 bytes of RAM, and print the returned pointer as a ”long”. Don’t delete it afterwards, to see what happens if you fill up the Arduino’s memory. — “new RAM” — Upon allocating 512 bytes of RAM, our program displays 1390 to the console, then 0 if we run again. If we change the allocated memory to 256 we were able to get multiple increments before the heap filled up.
- Read the pointer ”int *ptr=NULL;” — “null” — Returns the read pointer and respective value: 203?
- Use ”new” to allocate an array of size 3. Read from array element 100. — “new array” — Array returns an integer, which we assume to be a memory address (22304).

3 Conclusion

- **Storage Space:** 2020 bytes (Assignment 0) to 6022 bytes (Assignment 1) to 8002 bytes (Assignment 2).
- Assignment 2 is approximately 3.96 times larger (in terms of storage space) than Assignment 0.
- **Dynamic Memory:** 346 bytes (Assignment 0) to 614 bytes (Assignment 1) to 1287 bytes (Assignment 2).
- Assignment 2 is approximately 3.72 times larger (in terms of dynamic memory) than Assignment 0.

4 Appendix

4.1 Assignment 0: Source Code

```
// Benjamin Stream & Solomon Himelbloom
// Assignment 0

void setup() {
  Serial.begin(9600);
  Serial.print("Ready for commands (v0.1: help/whitespace)\n");
}

void loop() {
  while (Serial.available() != 0) {
    int c = Serial.read(); // check RX line
    // Print the received command.
    Serial.print("Command: {");
    Serial.print((char)c);
    Serial.print("} = 0x");
    Serial.print((int)c, HEX);
    Serial.print("\n");

    // "help" should print a brief summary
    // of the currently supported commands.
    if (c == 'p') {
      Serial.print("\n>> Help Menu (Lab 08.0):\n");
      Serial.print("ok: Checks if aurdino is alive\n");
      Serial.print("\\n or \\r: Adds a line of whitespace\n");
    }

    // "ok" should print "OK"
    // Useful for checking if the Arduino is still alive.
    if (c == 'k') {
      Serial.print("OK\n");
    }

    // Checks if \ (92 in ASCII) is inputed.
    if (c == 92) {
      if (c == 'n' || 'r') {
        Serial.print("\n");
      }
    }
  }

  delay(1000);
}
```

4.2 Assignment 1: Source Code

```
// Benjamin Stream & Solomon Himelbloom
// Assignment 1

void setup() {
  Serial.begin(9600);
  Serial.print("Ready for commands (v1.0)\n");
}

String buffer;


void clearBuffer() {

  buffer = "";

}
// Autoclears Buffer
void analogAnalysis(int pin) {
  int realPin = pin + 13;
  int v = analogRead(realPin); // raw 0-1023 analog-to-digital reading
  float V = 5.0 * v / 1023.0; // scale to float volts
  Serial.print("Raw Reading: ");
  Serial.print(v);
  Serial.print("\n");
  Serial.print("Voltage Reading: ");
  Serial.print(V);
  Serial.print("\n");
  clearBuffer();
}


void loop() {

  while (Serial.available())
  {
    char c = Serial.read();
    buffer += c;
    switch (c)
    {
      // Manual Buffer Clear (if it gets cluttered)
      case '*':
        Serial.print("Clearing Buffer...\n");
        Serial.print("Current Buffer:" + buffer + "\n");
        clearBuffer();
        Serial.print("Buffer Cleared!\n");

      default:
      // testing purposes can be disabed any time
      // Serial.print(buffer + "\n");

      case 'n':
        if (buffer == "\n") {
          Serial.print("NEWLINE\n");
          clearBuffer();
        }

      case 'r':
        if (buffer == "\r") {
          Serial.print("NEWLINEU\n");
          clearBuffer();
        }
```

```
63
64        case 'p':
65          if (buffer == "help") {
66            // "help" should print a brief summary
67            // of the currently supported commands.
68            Serial.print("Help Menu (Lab 08.1):\n");
69            Serial.print("ok: Checks if aurdino is alive\n");
70            Serial.print("\\n or \\r: Adds a line of whitespace\n");
71            Serial.print("A#: Checks analog ports raw data and voltage 0-5\n");
72            clearBuffer();
73          }
74
75        case 'k':
76          if (buffer == "ok") {
77            // \ok" should print "OK"
78            // Useful for checking if the Arduino is still alive.
79            Serial.print("OK\n");
80            clearBuffer();
81          }
82
83        case '0':
84          if (buffer == "A0") {
85            Serial.print("Analysis of A0\n");
86            analogAnalysis(c);
87          }
88
89        case '1':
90          if (buffer == "A1") {
91            Serial.print("Analysis of A1\n");
92            analogAnalysis(c);
93          }
94
95        case '2':
96          if (buffer == "A2") {
97            Serial.print("Analysis of A2\n");
98            analogAnalysis(c);
99          }
100
101       case '3':
102         if (buffer == "A3") {
103           Serial.print("Analysis of A3\n");
104           analogAnalysis(c);
105         }
106
107       case '4':
108         if (buffer == "A4") {
109           Serial.print("Analysis of A4\n");
110           analogAnalysis(c);
111         }
112
113       case '5':
114         if (buffer == "A5") {
115           Serial.print("Analysis of A5\n");
116           analogAnalysis(c);
117         }
118     }
119   }
120   delay(1000);
121 }
```

4.3 Assignment 2: Source Code

```
// Benjamin Stream & Solomon Himelbloom
// Assignment 2

void setup() {
  Serial.begin(9600);
  Serial.print("Ready for commands (v2.0)\n");
  Serial.print("*: Clears buffer.\n");
}

String buffer;
String hexBuffer;

void clearBuffer() {
  buffer = "";
}

// Autoclears Buffer
void analogAnalysis(int pin) {
  int realPin = pin + 13;
  int v = analogRead(realPin); // raw 0-1023 analog-to-digital reading
  float V = 5.0 * v / 1023.0; // scale to float volts
  Serial.print("Raw Reading: ");
  Serial.print(v);
  Serial.print("\n");
  Serial.print("Voltage Reading: ");
  Serial.print(V);
  Serial.print("\n");
  clearBuffer();
}


void loop() {

  while (Serial.available())
  {
    char c = Serial.read();
    buffer += c;
    switch (c)
    {
      // Manual Buffer Clear (if it gets cluttered)
      case '*':
        Serial.print("Clearing Buffer...\n");
        Serial.print("Current Buffer:" + buffer + "\n");
        clearBuffer();
        Serial.print("Buffer Cleared!\n");

      case 'y':
        if (buffer == "new array") {
          // new array size[3] then read array[100]
          Serial.print("Array Magic!\n");
          int magicalArray[3];
          Serial.print(magicalArray[100]);
          Serial.print(": Wow isnt that magical?\n");
          clearBuffer();
        }

      case 'l':
        if (buffer == "null") {
          // Read the pointer "int *ptr=NULL;"
          Serial.print("Reading the null pointer.\n");
```

```
61            int *ptr = NULL;
62
63            Serial.println(*ptr);
64
65            Serial.print("End of command (null).\n");
66            clearBuffer();
67          }
68
69        case 'M':
70          if (buffer == "new RAM") {
71            // Allocate 512 bytes of ram to the heap
72            Serial.print("Allocating 512 bytes RAM!\n");
73            unsigned char* ramAllocate = new unsigned char[512];
74            long ptr_to_ramAllocate = ramAllocate;
75            Serial.print("Allocation done.\n");
76            Serial.print(ptr_to_ramAllocate);
77            Serial.print("\n");
78            clearBuffer();
79          }
80
81        case 'n':
82          if (buffer.startsWith("\\")) {
83            Serial.print("\n");
84            clearBuffer();
85          }
86
87        case 'r':
88          if (buffer.startsWith("\\")) {
89            Serial.print("\n");
90            clearBuffer();
91          }
92
93        case 'p':
94          if (buffer == "help") {
95            // "help" should print a brief summary
96            // of the currently supported commands.
97            Serial.print("\nHelp Menu (Lab 08.2):\n");
98            Serial.print("*: Clears buffer.\n");
99            Serial.print("ok: Checks if aurdino is alive.\n");
100           Serial.print("\\n or \\r: Adds a line of whitespace\n");
101           Serial.print("A#: Checks analog ports raw data and voltage 0-5.\n");
102           Serial.print("on: Turn the pin 13 debug LED on.\n");
103           Serial.print("off: Turn the pin 13 debug LED off.\n");
104           Serial.print("x: Print 'x' to the serial port 10000 times.\n");
105           Serial.print("baud: Change the serial baud rate to 19200.\n");
106           Serial.print("grüning: Print the characters in hex.\n");
107           Serial.print("new RAM: Allocates 512 bytes to memory.\n");
108           Serial.print("null: Read the pointer int *ptr=NULL;.\n");
109           Serial.print("new array: Does assignment task.\n");
110           clearBuffer();
111         }
112
113       case 'k':
114         if (buffer == "ok") {
115           // "ok" should print "OK"
116           //  Useful for checking if the Arduino is still alive.
117           Serial.print("OK\n");
118           clearBuffer();
119         }
120
121       case '0':
122         if (buffer == "A0") {
123           Serial.print("Analysis of A0\n");
```

```
124             analogAnalysis(c);
125           }
126
127         case '1':
128           if (buffer == "A1") {
129             Serial.print("Analysis of A1\n");
130             analogAnalysis(c);
131           }
132
133         case '2':
134           if (buffer == "A2") {
135             Serial.print("Analysis of A2\n");
136             analogAnalysis(c);
137           }
138
139         case '3':
140           if (buffer == "A3") {
141             Serial.print("Analysis of A3\n");
142             analogAnalysis(c);
143           }
144
145         case '4':
146           if (buffer == "A4") {
147             Serial.print("Analysis of A4\n");
148             analogAnalysis(c);
149           }
150
151         case '5':
152           if (buffer == "A5") {
153             Serial.print("Analysis of A5\n");
154             analogAnalysis(c);
155           }
156
157         case 'o':
158           // Turns the LED on.
159           if (buffer == "on") {
160             pinMode(13, OUTPUT);
161             digitalWrite(13, 1);
162             Serial.print("LED: [ON]\n");
163             clearBuffer();
164           }
165
166         case 'f':
167           // Turns the LED off.
168           if (buffer == "off") {
169             pinMode(13, OUTPUT);
170             digitalWrite(13, 0);
171             Serial.print("LED: [OFF]\n");
172             clearBuffer();
173           }
174
175         case 'x':
176           // Print "x" to the serial port.
177           if (buffer == "x") {
178             int i;
179             Serial.print("-----BEGIN SERIAL PORT MESSAGE-----\n");
180             for (int i = 0; i < 9999; i++) {
181               Serial.print("x");
182             }
183             Serial.print("\n-----END SERIAL PORT MESSAGE-----\n");
184             clearBuffer();
185           }
186
187         case 'd':
188           // Change the serial baud rate to 19200.
```

```
        if (buffer == "baud") {
          Serial.flush();
          delay(10);
          Serial.begin(19200);
          Serial.print("Are you reading this?\n");
          Serial.flush();
          delay(10);
          Serial.begin(9600);
          clearBuffer();
        }

      default:

      case 'g':
        // Test command names with Unicode chars.
        if (buffer == "grüning") {
          Serial.print("grüning\n");
          Serial.print("Print the chars in hex.\n");
          Serial.print("{");
          Serial.print(buffer);
          Serial.print("} = 0x");
          for (int i = 0; i < (int)buffer.length(); i++) {
            unsigned char hexChar = buffer[i];
            Serial.print((int)hexChar, HEX);
          }
          Serial.print("\n");
          clearBuffer();
        }
    }
  }
  delay(1000);
}
```