The GCC compiler treats the general purpose registers in three distinct ways. List the ways and indicate what is unique about each.

Demo check-off: _____

Complete the C function below:

```
#include <avr/io.h>

/**
  * Write the sum of the values on PORTA and PORTB to PORTC.
  *
  * The function does not assume that any of the ports have been configured
  * prior to being called.
  * You may assume that the values found on PORTA and PORTB are unsigned
  * integers whose sum is less than 255.
  */
void a_plus_b_written_to_c()
{









}
```

Complete the C function below:

```
/**
  * Wait until SW1 on the keypad has been depressed and then return.
  *
  * In order to work properly, the keypad must be connected to PORTA.
  * The function does not assume that PORTA has been configured prior to
  * being called.
  *
  * This function may assume that only one key on the keypad is pressed
  * at a time.
  */
void wait_for_sw1()
{









}
```

(a) Explain why the SunRom boards need a separate chip to handle serial communication via an RS-232 connection.

(b) Draw and describe one data frame used to accomplish serial communication via the USART subsystem. Indicate what each bit in the data frame represents.

Write the header file, Student.h, that will allow the following code to compile.

```
#include "Student.h"

int main()
{
  Student chris;
  Student *ptr = &chris;

  ptr->id = 12345;
  chris.gpa = 3.98;

  return 0;
}
```

Consider the rectangle.h file (on back of this page).
(a) Write the line(s) of code necessary to create two Rectangles each with one corner at (100,100) and the other at (200,200). One Rectangle should be allocated on the stack and the other on the heap.

(b) What does the * in the (*getArea) signify?

(c) Why do we pass a Rectangle pointer to the moveTo function?

```
#ifndef RECTANGLE_H
#define RECTANGLE_H

#include "RectanglePrivate.h"

typedef struct {
  int x;
  int y;
} Point;

typedef struct _Rectangle Rectangle;

struct _Rectangle {
  Point ul;
  Point lr;

  int (*getArea)(const Rectangle const *this);
  int (*getPerim)(const Rectangle const *this);
  void (*moveTo)(Rectangle *this, const Point to);
};

void RectangleCtr(Rectangle *this, int ulx, int uly, int lrx, int lry);

Rectangle* createRectangle(int ulx, int uly, int lrx, int lry);

#endif
```

Implement the following function in assembly:

```
/**
 * Silly function to test students' ability to mix C and assembly code.
 *
 * @param for_ddrb Value to be sent to DDRB.
 * @param for_portb Value to be sent to PORTB.
 * @return a 16-bit value with for_ddrb in the upper byte
 *         and for_portb in the lower byte.
 */
extern uint16_t quiz7(uint8_t for_ddrb, uint8_t for_portb);
```

The .s file has been started for you:

```
#define __SFR_OFFSET 0
#include <avr/io.h>

.global quiz7

.section .text
```

Consider the rectangle.h file developed in lecture (on back of this page).
(a) Write the line(s) of code necessary to create a Rectangle with one corner at (100,100) and the other at (200,200).

(b) What does the `static` keyword in front of the last function prototype do?

(c) Why is the `static` keyword placed in front of the last function prototype?

```
typedef struct {
  int x;
  int y;
} Point;

typedef struct {
  Point ul;
  Point lr;

  int (*getArea)(const struct Rectangle const *this);
  int (*getPerim)(const struct Rectangle const *this);
  void (*moveTo)(struct Rectangle *this, const Point to);
} Rectangle;

void _Rectangle(Rectangle *this, int ulx, int uly, int lrx, int lry);

static int getArea(const Rectangle const *this);

static int getPerim(const Rectangle const *this);

static void moveTo(Rectangle *this, const Point to);
```