1. The **due date** for this assignment is Wednesday, November 3.

2. Your submission must be made in class on the due date, and must have the course cover-sheet as the first page. The cover sheet is available online at the course URL: www.cs.uml.edu/~bill/cs520/Assignment_Cover.pdf

3. You must submit **hardcopy** of your **source code**, **your output** and a **write-up** of your results described below. The course cover-sheet should be the first page of your submission.

4. For this project we have a Fedora Core 6 linux system that can be reached from our mercury system by using the ssh command from mercury as:

```
ssh -l <your user name> 129.63.8.27
```

and then providing your password. Each student has an account with a user name that is the first 8 characters of their last name or just their last name (if <= 8 characters) with no upper case letters and an initial password set to **cs520**, so a student named John Smithworth would use a username of **smithwor** and a password of **cs520** to log in (notice that usernames are always in all lower case letters … see the last page of this handout for the class list).

5. Do not attempt to log into any account but your own, and the first time you log into this system, you should **change your password to something you can remember** (use the passwd command). You may create whatever directories you need under your login directory, but please stay out of directories that don’t belong to you. **(Remember, cheating is an offense that can result in an F for this course and your permanent expulsion from UML.)**

6. This system has a number of **scsi devices** attached to a parallel scsi bus, each of which is represented by a **generic scsi device node** that is located in the **/dev** directory with a name that is prefixed by **sg** (e.g. you will find the entries **/dev/sg0**, **/dev/sg1**, etc.)

7. For this assignment, you will have to work with these devices using the **scsi generic driver interface** that is provided in linux, and about which you have read in your last class reading assignment.

8. For each **enumerated sg device**, you must:
• Open the sg device using the `open()` system call with an `O_RDWR` channel flag

• Use the `SG_IO ioctl()` system call to retrieve the inquiry information available from each of the scsi devices on the system (i.e. each of the `sg_n` objects found on the system), and provide the following information for each object:
  - The scsi peripheral device type (byte 1 of return data)
  - If the object uses removable media (byte 2 of return data)
  - The Vendor name (starts at byte 8 of return data)
  - The Product ID (starts at byte 16 of return data)
  - The Product Revision Num (starts at byte 32 of return data)

9. One of the devices on this machine’s scsi bus is a disk device, and this device has a readable text message located at absolute block number 128. You must read this data from the disk device using the sg interface only, and then print it to standard out. The data include a list of entries that consist of a student name and a disk block number that the named student will have to work with for the next operation. There is an entry in the list for every registered student.

10. Find your entry in the class list, and read the disk block number assigned to you using the sg interface only (as you did for the previous step). There will be a message in this disk block that you must print and include in your report, and that you must use for the next step in the assignment.

11. Using the information that you found in the previous step, you must now write a three line message to the disk block that the previous message specified, using the sg interface only. The three line message must include:
   - Your full name
   - Your email address
   - The number of this disk block you are writing

12. You must provide the output of your program(s), showing the required inquiry data, and showing the text messages that you retrieved from the scsi disk device at disk block 128 and then at the block assigned specifically to you. The three line message that you should have written will be checked by me when I grade your assignment, so make sure you get it written in the correct location.
13. Your write-up should discuss your implementation, and should discuss how you have **proved that your implementation works** as required. If you have **problems** with your work you must discuss them here.

14. **Be careful with your attempts to write to the disk, since you could easily destroy the data that other students depend upon. Write only to your specific disk block, and only one sector or less worth of total information.**

The sg interface program should include:
```
#include <sys/types.h>
#include <scsi/scsi.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <sys/ioctl.h>
#include <scsi/sg.h>
```

and the principal data structure required to work with the sg interface (as you should have discovered in your readings) is the **sg_io_hdr**:

```c
typedef struct sg_io_hdr
{
    int interface_id;       // [i] 'S' for SCSI generic (req)
    int dxfer_direction;    // [i] data transfer direction
    unsigned char cmd_len;  // [i] SCSI command len <= 16 bytes
    unsigned char mx_sb_len; // [i] max length to write to sbp
    unsigned short iovec_count; // [i] 0 implies no scatter gather
    unsigned int dxfer_len;  // [i] byte count of data transfer
    void * dxferp;           // [i] [*io] points to data transfer
                             // memory or scatter-gather list
    unsigned char * cmdp;    // [i] [*i] points to SCSI command
    unsigned char * sbp;     // [i] [*o] points to sense_buffer
    unsigned int timeout;    // [i] MAX_UINT->no timeout (ms)
    unsigned int flags;      // [i] 0 -> default, see SG_FLAG...
    int pack_id;             // [i->o] unused internally
    void * usr_ptr;          // [i->o] unused internally
    unsigned char status;    // [o] scsi status
    unsigned char masked_status; // [o] shifted, masked scsi status
    unsigned char msg_status; // [o] messaging level data
    unsigned char sb_len_wr; // [o] byte count written to sbp
    unsigned short host_status; // [o] errors from host adapter
    unsigned short driver_status; // [o] errors from software driver
    int resid;                // [o] dxfer_len - actual bytes
    unsigned int duration;    // [o] time taken (unit: millsec)
    unsigned int info;        // [o] auxiliary information
} sg_io_hdr_t;          // around 64 bytes long (on i386)
```
User accounts for the system we are using are shown below, using the first 8 characters of the last name or just the last name (if <= 8 characters) with no upper case letters. **All names are in lower case letters only:**

harris
lu