We are 183

L18: Week 11 – Monday

major/minor expo

WEDNESDAY

March 23

11am to 3pm

ROGEL BALLROOM, MICHIGAN UNION

FREE SWAG!

explore the possibilities

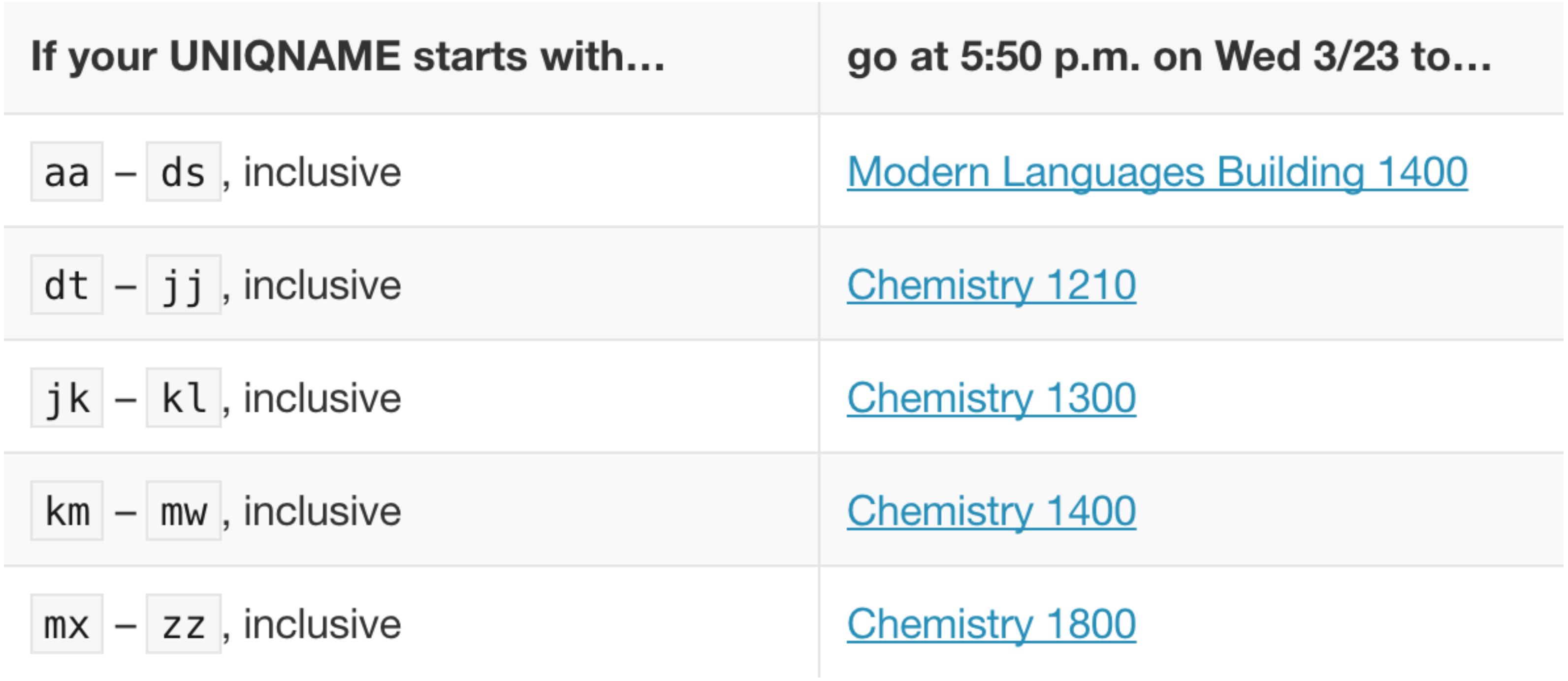
explore

M | LSA NEWNAN ACADEMIC ADVISING CENTER
UNIVERSITY OF MICHIGAN

#exploreLSA explore.lsa.umich.edu

Reminders

- Exam 2 is on Wednesday!



If your UNIQNAME starts with…	go at 5:50 p.m. on Wed 3/23 to…
`aa` – `ds`, inclusive	Modern Languages Building 1400
`dt` – `jj`, inclusive	Chemistry 1210
`jk` – `kl`, inclusive	Chemistry 1300
`km` – `mw`, inclusive	Chemistry 1400
`mx` – `zz`, inclusive	Chemistry 1800

- Final Project Proposal due Friday

Exam Preparation

EECS 183, W'16

Exam Taking Tips

- Start with the Multiple Choice portion
  - Do the easy ones, build your confidence
  - Mark those you need to come back to
- Move on to the Free Response portion
  - Less rush for time
  - Do the easy ones and come back to the others
- Finish up both portions

- Don't spend too much time on one question
  - Skip it, revisit later

Exam Taking Tips

- Read each question thoroughly and carefully
- Take a quick break between each question
  - Close your eyes
  - Take a deep breath
  - Reset your mind
- Finish reading the question before you start thinking of the answer
- Take a few seconds to think about the answer before you write anything down or look at the answer options

Free Response Practice

- Best practice:
  - Put away your computer
  - Write out your answer on a sheet of paper
- This is the best practice when studying past exams as well
  - Print out past exams
  - Write out answers and time yourself

Practice Question #1

We want to write a function that will draw a Z shape from a square 2D array (i.e. a 2D array with an equal number of rows and columns).

Examples:

```
size 3:
1 1 1
0 1 0
1 1 1
```

```
size 4:
1 1 1 1
0 0 1 0
0 1 0 0
1 1 1 1
```

```
size 5:
1 1 1 1 1
0 0 0 1 0
0 0 1 0 0
0 1 0 0 0
1 1 1 1 1
```

Practice Question #1

Implement the following function according to its RME.

```
/**
 * Requires: size > 0 and size <= MAX_SIZE
 * Modifies: board
 * Effects: Sets all of the elements in the first
 *   row, last row, and top-right-to-bottom-left
 *   diagonal to true, and sets all other elements
 *   to false (within bounds of size rows and size
 *   columns)
 */
void formZ(bool board[MAX_SIZE][MAX_SIZE], int size);
```

Practice Question #1

```
void formZ(bool board[MAX_SIZE][MAX_SIZE], int size){
    for (int row = 0; row < size; ++row) {
        for (int col = 0; col < size; ++col) {
            if (row == 0 || row == (size – 1)) {
                board[row][col] = true;
            } else if (row + col == (size – 1)) {
                board[row][col] = true;
            } else {
                board[row][col] = false;
            }
        }
    }
}
```

Practice Question #2

We are volunteering at a local senior center and want to help the residents keep track of their medication. We can use the class:

```
class Medication {
private:
    string name;
    int dose_mg;
public:
    Medication();
    Medication(string med_name, int dose);
    void change_dose(int new_dose);
    string get_name();
    int get_dose();
};
```

Write each of the member functions, as if in Medication.cpp.

Practice Question #2

```
Medication::Medication() {
    name = "";
    dose_mg = 0;
}
Medication::Medication(string med_name, int dose) {
    name = med_name;
    dose_mg = dose;
}
void Medication::change_dose(int new_dose) {
    dose_mg = new_dose;
}
string Medication::get_name() {
    return name;
}
int Medication::get_dose() {
    return dose_mg;
}
```

Practice Question #3

Now we want to help the residents organize their medication by time of day that they need to take it. We can use the class:

```
const int MAX_MEDS = 10;
class MedSchedule {
private:
    Medication morning[MAX_MEDS];
    Medication night[MAX_MEDS];
    int num_morning;
    int num_night;
public:
    MedSchedule();
    MedSchedule(Medication m[], Medication n[], int num_m, int num_n);
    int mg_daily(string med_name);
};
```

Practice Question #3

Here's the RME for the last function:

```
class MedSchedule {
…
    /**
     * Requires: nothing
     * Modifies: nothing
     * Effects: Returns the number of mg of the Medication
     *   specified by med_name that is taken each day, i.e.
     *   the combined doses of morning and night, if it is
     *   taken at those times.
     */
    int mg_daily(string med_name);
};
```

Write the implementation of the function, as in MedSchedule.cpp.

Practice Question #3

```
int MedSchedule::mg_daily(string med_name) {
    int mg = 0;
    for (int i = 0; i < num_morning; ++i) {
        if (morning[i].get_name() == med_name) {
            mg += morning[i].get_dose();
        }
    }
    for (int i = 0; i < num_night; ++i) {
        if (night[i].get_name() == med_name) {
            mg += night[i].get_dose();
        }
    }
    return mg;
}
```

Practice Question #4

Now write a main function to initialize an instance of the MedSchedule class for a person that takes:

- Lipitor – 10mg once per day at night
- Vitamin D – 1000mg twice per day
- Multi-Vitamin – 50mg once per day in morning

Print out the number of mg of Vitamin D that the person takes daily.

Practice Question #4

```
int main() {
    Medication morning[2];
    morning[0] = Medication("Vitamin D", 1000);
    morning[1] = Medication("Multi-Vitamin", 50);
    Medication night[2];
    night[0] = Medication("Vitamin D", 1000);
    night[1] = Medication("Lipitor", 10);
    MedSchedule schedule(morning, night, 2, 2);
    cout << schedule.mg_daily("Vitamin D");
    return 0;
}
```