Plan 9

November 6 – November 13

Read Section 12.1 – 12.5 in the textbook.

- **Exercise 13** Solve the exercise on the next pages.

Exercise 13 The C function below is an implementation of insertion sort for an array of integers.

```
void sort(int a[], int n) {
    int i, j, v;
    for (i = 1; i < n; i++) {
        v = a[i];
        for (j = i; j > 0 && v < a[j - 1]; j--)
            a[j] = a[j - 1];
        a[j] = v;
    }
}
```

(a) Generate x86 assembly code for this function.

(b) Optimize the code by keeping the values of `i`, `j` and `v` in registers (thereby reducing the number of memory references).

(c) Test your code using the C program on the next two pages.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <assert.h>

void sort(int a[], int n);

void shuffle(int a[], int n) {
    int i, j, temp;
    for (i = 1; i < n ; i++) {
        j = rand() % (i + 1);
        temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
}
```

cont'd on next page

```
int main() {
    int N = 100000, start_time, i;
    int *a = (int *) malloc(N * sizeof(int));
    for (i = 0; i < N; i++)
        a[i] = i;
    srand(7913);
    shuffle(a, N);
    start_time = clock();
    sort(a, N);
    printf("Time used = %0.2f seconds\n",
        (float) (clock() - start_time) / CLOCKS_PER_SEC);
    for (i = 0; i < N; i++)
        assert(a[i] == i);
}
```