Weekplan: Warmup

Philip Bille

Reading

Course Guide and Programming Preparation.

Exercises

1 Loops The purpose of this exercise is to check if you understand some basic programming concepts. You should *not* implement the functions. What do the Java/C/C++ functions `loop1`, `loop2`, `loop3` and `loop4` in Figure 1 return when

1.1 `n` $= 4$?

1.2 `n` $= 10$?

1.3 `n` $= 1000$?

1.4 as a function of `n`?

2 Recursion and Iteration A function is *recursive* if it calls itself. For instance the Java/C/C++ function `f(A,n)` in Figure 1 is recursive (to obtain a valid C/C++ program `[]` should be changed to `*`). Solve the following exercises.

2.1 What does `f(A,n)` compute if `A` is an array of integers of length `n`? You should *not* implement it.

2.2 Rewrite `f(A,n)` to be *iterative*, ie. make a function that computes the same as `f(A,n)` but without calling itself.

3 Introduction to CodeJudge and Piazza Solve the following exercises.

3.1 Go to CodeJudge and read the tutorial.

3.2 [$w\dagger$] Implement a function to add two numbers and test it in CodeJudge.

3.3 Log on Piazza.

4 Linearthritis You have recently hired 128 programmers to your new high-tech startup company. Unfortunately, one of them is suffering from the feared Linearthritis disease that makes everybody near the person write slow programs. In order to identify the diseased programmer you have rented a special room that you can use to determine if the diseased programmer is with in a group of your programmers. It is extremely expensive to rent this room and the process needed to test a group is complicated (long and exhausting programming tests are necessary). Therefore you would like to minimize the number of times you have to use the room in order to find the diseased. Solve the following exercises.

4.1 Show you can find the diseased programmer using a most 7 tests.

4.2 How many tests do you need if you have n programmers instead of 128?

4.3 [$*$] Assume you rent $k > 1$ rooms you can use to test k groups of programmers simultaneously. How many *rounds* of tests are enough to identify the diseased programmer? In each round you can test k groups in parallel.

```
int loop1(int n) {
   int x = 0;
   for(int i = 0; i < n; i++) {
      for(int j = 0; j < n; j++) x++;
   }
   return x;
}

int loop2(int n) {
   int x = 0;
   for(int i = 0; i < n; i++) x++;
   for(int j = 0; j < n; j++) x++;
   return x;
}

int loop3(int n) {
   int x = 0;
   for(int i = 0; i < n; i++) {
      if (i == n-1) for(int j = 0; j < n; j++) x++;
   }
   return x;
}

int loop4(int n) {
   int x = 0;
   for(int i = 0; i < n; i++) {
      for(int j = i; j < n; j++) x++;
   }
   return x;
}

int f(int[] A, int n) {
   if(n == 0) return 0;
   else return f(A, n - 1) + A[n-1];
}
```

Figure 1: Loops and recursion.

5 Zombie Duels You have an army of n brainless zombies. You want to find the strongest and the weakest zombie in the army. By pairing up two zombies in a cage with a big hunk of meat you can quickly determine who of the two are the strongest. Unfortunately it tears down the zombies to duel, so you want to minimize the number of duels needed. Solve the following exercises.

5.1 Explain how to find the strongest zombie using at most $n-1$ duels.

5.2 [$*$] Explain how to find both the strongest and the weakest zombie using at most $3n/2$ duels.

5.3 [$**$] Explain how to find both the strongest and second strongest zombie using at most $n + \log_2 n$ duels.