Informatics II, Spring 2023, Exercise 2

Publication of exercise: February 27, 2023
Publication of solution: March 5, 2023
Exercise classes: March 6 - 10, 2023

Task 1

1.1 Output: true
The program tests if a string (array of characters) is a palindrome i.e. it uses recursion to check if a word is read the same forward and backwards.

1.2 $O(n)$

Task 2

2.1 Output: 00101

2.2 $20\%2 = 0$; `printf(0); printRec(20)`
↪ $10\%2 = 0$; `printf(0); printRec(10)`
↪ $5\%2 = 1$; `printf(1); printRec(5)`
↪ $2\%2 = 0$; `printf(0); printRec(2)`
↪ $1\%2 = 1$; `printf(1); printRec(1)`
↪ `printRec(0)`
terminating condition

2.3 Answer: Swap lines 6 and 8

```
1 void printRec(int n)
2 {
3   if (n == 0)
4     return;
5
6   printRec(n/2);
7
8   printf("%d", n%2);
9 }
```

Task 3

```
int p[3]; // three pegs

int N; // number of disks

int M; // counter for number of moves
int K; // number of state to be displayed (-1 = all states)

// ------------------------------------------------------------------------

void print_disk(int d, char c) {
  int k;
  for (k = 0; k < N-d+1; k++) { printf("␣"); }
  for (k = 0; k < 2*d-1; k++) { printf("%c", c); }
  for (k = 0; k < N-d+1; k++) { printf("␣"); }
}

// ------------------------------------------------------------------------

void print_pegs() {
  int i, j, l;
  int t[3] = { p[0], p[1], p[2] };
  if (M != K && K != -1) { return; }
  printf("Tower␣of␣Hanoi␣pegs␣after␣%d␣moves:\n", M);
  j = pow(10, N-1);
  for (l = 0; l < N; l++) {
    for (i = 0; i < 3; i++) {
      if (t[i] / j != 0) {
        print_disk(t[i]%10, '*');
        t[i] = t[i] / 10;
      } else {
        print_disk(1, '␣');
      }
    }
    j = j / 10;
    printf("\n");
  }
}

// ------------------------------------------------------------------------

void Hanoi(int n, int A, int B, int C) {
  if (n == 0) return;
  Hanoi(n-1, A, C, B);
  p[B] = p[B] * 10 + p[A] % 10;
  p[A] = p[A] / 10;
  M++;
  print_pegs();
  Hanoi(n-1, C, B, A);
}

// ------------------------------------------------------------------------

int main(int argc, char **argv) {
  sscanf(argv[1], "%d", &N);
  sscanf(argv[2], "%d", &K);

  int m = 1;
  p[1] = 0;
  p[2] = 0;
  for (int i = 0; i < N; i++) {
    p[0] = p[0] + (i + 1) * m;
```

```
62    m = m * 10;
63  }
64
65  M = 0;
66  print_pegs();
67  Hanoi(N, 0, 1, 2);
68 }
```

Task 4

```
1  void drawTriangle(int A[], int n, int level) {
2     if (n < 1) {
3        return;
4     }
5
6     int t[n-1];
7     for (int i = 0; i < n-1; i++) {
8        t[i] = A[i] + A[i+1];
9     }
10    drawTriangle(t, n-1, level+1);
11
12    for (int j = 0; j <= level; j++) {
13       printf("␣");
14    }
15    for (int j = 0; j < n; j++) {
16       if(A[j]<10){
17          printf("%d␣␣", A[j]);
18       }else{
19          printf("%d␣", A[j]);
20       }
21    }
22    printf("\n");
23 }
24    drawTriangle(A, n, 1);
```