National University of Singapore
School of Computing
CS1010X: Programming Methodology
Semester II, 2019/2020

Tutorial 11
C Programming

1. A skeleton of a C implementaion of Towers of Hanoi has been provided for you:

```
#include <stdio.h>
#include <string.h>
#define NUM_DISCS 4

typedef struct {
    int peg[3][ ... ];
    char name[30];
} Towers;

void make_moves(Towers *t, int n, int src, int des, int aux);
void init_towers(Towers *t, const char name[]);
void print_towers(Towers t);
void move_disc(Towers *t, int src, int des);

int main(void) {
    Towers t;
    init_towers(&t, "Hanoi");
    printf("Towers of %s\n", t.name);
    print_towers(t);
    make_moves(&t, NUM_DISCS, 0, 2, 1);
    return 0;
}

void make_moves(Towers *t, int n, int src, int des, int aux) {
    if (n<=0) return;
    make_moves(t, n-1, src, aux, des);
    move_disc(t, src, des);
    make_moves(t, n-1, aux, des, src);
    return;
}

void move_disc(Towers *t, int src, int des) {
    ...
    print_towers(*t);
    return;
}

...
```

Running the program generates the following output:

```
[ 0 1 2 3 ], [ ], [ ]
[ 0 1 2 ], [ 3 ], [ ]
[ 0 1 ], [ 3 ], [ 2 ]
[ 0 1 ], [ ], [ 2 3 ]
[ 0 ], [ 1 ], [ 2 3 ]
[ 0 3 ], [ 1 ], [ 2 ]
[ 0 3 ], [ 1 2 ], [ ]
[ 0 ], [ 1 2 3 ], [ ]
[ ], [ 1 2 3 ], [ 0 ]
[ ], [ 1 2 ], [ 0 3 ]
[ 2 ], [ 1 ], [ 0 3 ]
[ 2 3 ], [ 1 ], [ 0 ]
[ 2 3 ], [ ], [ 0 1 ]
[ 2 ], [ 3 ], [ 0 1 ]
[ ], [ 3 ], [ 0 1 2 ]
[ ], [ ], [ 0 1 2 3 ]
```

The functions have been left uncompleted:

- `init_towers` initializes the `Tower` so that it represent the starting configuration of a game.
- `print_towers` prints the given `Tower` in the above format.
- `move_disc` moves the topmost disc from source peg `src` to destination peg `des`, then prints the resulting state using `print_towers`.

Tasks

(a) Describe how you would use the structure `Tower` to represent a Towers of Hanoi game. What size should you initialize the array representing each peg to? Why?

Hint: Use a sentinel value to indicate the location of the last disc of each peg.

(b) Provide an implementation for `init_towers` and `print_towers`.

(c) Provide an implementation for `move_discs`.

2. The letter at the end of a National Registration Identity Card (NRIC) number is called its *check code*. As its name suggests, the check code allows us to check if a NRIC number has been entered correctly. A typographical error, for example, will result in a mismatched check code. The algorithm for generating the check code is as follows:

   1. Obtain the weighted sum of the NRIC digits using the weights $< 2, 7, 6, 5, 4, 3, 2 >$. For the NRIC number 9300007, the sum is

   $$(9 * 2) + (3 * 7) + (0 * 6) + (0 * 5) + (0 * 4) + (0 * 3) + (7 * 2) = 53$$

   2. Find the remainder of the sum when divided by 11: $53 \ \% \ 11 = 9$.
   3. Substract the remainder from 11: $11 - 9 = 2$.
   4. Look the check code up:

1	**2**	3	4	5	6	7	8	9	10	11
A	**B**	C	D	E	F	G	H	I	Z	J

   The check code for 9300007 is B.

Task

Write a function `char get_check_code(int [])` that takes in an array containing the seven digits of an NRIC number, and generates its check code.

Sample execution runs:

```
Enter your 7-digit NRIC number: 9300007
Your check code is 'B'.
```

```
Enter your 7-digit NRIC number: 1245133
Your check code is 'D'.
```