Exercise 1.1

1. Install Spin and iSpin by following steps 0–2 on [http://spinroot.com/spin/Man/README.html](http://spinroot.com/spin/Man/README.html).

2. Inspect contents of the downloaded package. It should contain several examples and documents to start with. To test your installation, run the following commands in the Examples directory:
   - `spin --`
   - `spin -V`
   - `spin hello.pml`
   - `ispin hello.pml`

   Spin references can be downloaded from [http://spinroot.com/spin/Man/](http://spinroot.com/spin/Man/) (For a gentle introduction to Spin, see e.g. Tutorial_1.pdf)

3. Install Modex from [http://spinroot.com/modex/](http://spinroot.com/modex/). Modex is a tool that can extract Spin models from programs written in the C programming language.

4. To test your installation, run the following commands in the Manual directory:
   - `modex --`
   - `modex hello.c`
   - `spin model`

5. Compare the contents of `hello.pml` and `model`.

6. In the Modex package, there is a script named `verify`. Given a C program, the script calls Modex and Spin, and outputs user-friendly messages. Copy the script or make a link to it in the bin directory. For instance,
   - `cp Scripts/verify /usr/local/bin`

7. To test the script, run:
   - `verify hello.c`  # perform model extraction + verification
   - `verify clean`    # clean up temporary files
Exercise 1.2

Consider the following program `bounds.c`:

```c
#define N 3
#define M N+1

int main(void) {
  int *p[N][M], q[N*M], i, j, k = 0;

  for (i = 0; i < N; i++) {
    for (j = 0; j < M; j++) {
      p[i][j] = &q[k];
      k++;
    }
  }
}
```

1. Can you spot a bug in the program? Justify your answer.

2. Run Modex and Spin to find the bug. Observe the output messages.

3. Inspect the content of the generated model file.

Exercise 1.3

Consider the following program `threads.c` (an example from the Modex distribution):

```c
#include <pthread.h>
#include <assert.h>

int shared = 0;
int *ptr;

void *thread1(void *arg) {
  int tmp;
  return 0;
}

void *thread2(void *arg) {
  int tmp;
  return 0;
}

int main(void) {
  pthread_t t[2];
  pthread_create(&t[0], 0, thread1, 0);
  pthread_create(&t[1], 0, thread2, 0);
  pthread_join(t[0], 0);
  pthread_join(t[1], 0);
}
```
1. Does the assertion at line 37 always hold? Justify your answer.

2. Run Model and Spin or verify to confirm your finding.
Solution 1.2

1. \#define M N+1 is the problematic line. The C compiler replaces all instances of M with N+1 without any parenthesis. Hence, the size of q would be N*(N+1) instead of N*(N+1).

2. Run modex bounds.c and spin -a model. This creates the pan.c file. Next compile it and execute it gcc -o pan pan.c && ./pan. You would get an error which says the following: pan:1: c_code line 26 precondition false: (Pp_main->k < ((3*3)+1)) (at depth 52)

3. The model file has a line c_state "int q[((3*3)+1)]" "Local p_main" which gives away the problem.

Solution 1.3

1. No, it does not hold. Consider the following execution sequence after both the threads are created: lines 8, 10, 11 (thread1.tmp = 0), 18, 20, 21 (thread2.tmp = 0), 22 (thread2.tmp = 1), 23 (shared = thread2.tmp = 1), 25, 12 (thread1.tmp = 1), 13 (shared = thread1.tmp = 1).

2. On running ./pan, we get the following error pan:1: c_code line 91 precondition false: (now.shared==2) (at depth 35)