1. Design test cases for the following program with the “simple loop” strategy:

```c
x=0; read(y);
while ((y > 100) && (x < 10)) { x=x+1; read(y); }
print(y);
```

2. Let us test the following program.

```c
x=0; read(y);
while (y > x) { x=x+y; read(y); }
if (x < 100) print("small") else print("large");
```

a) Construct a data-flow graph for the program with respect to variable \( x \).
b) Which execution paths have to be traversed during testing, in order to reach complete \textit{all-definitions} coverage with respect to variable \( x \)? Minimize the number of paths and tests.
c) Which execution paths have to be traversed during testing, in order to reach complete \textit{all-uses} coverage with respect to variable \( x \)? Minimize the number of paths and tests.
d) Design test cases for reaching the (minimal) complete \textit{all-uses} coverage with respect to variable \( x \).

3. Analyze the practical relevance of white-box testing methods. In what kind of situations is (a) statement coverage, (b) branch coverage, (c) multicondition coverage, (d) all-uses coverage better than the other coverage-based methods? Which one (if any) is your favorite?

4. Let us study the testing of the Mealy model of a two-player video game, given as a state machine in Chapter 6 of the lecture slides of the course.

a) Design test cases for reaching the \textit{all-states} coverage over the state machine.
b) Design test cases for reaching the \textit{all-transitions} coverage over the state machine.