i = 0;

while (i < n) {

j = 0;

while (j < m) {

if (i < j) {

val = val + i;

} else if (j == i) {

val = val - 1;

} else { // j > i

val = val + j;

}

j = j + 1;

}

i = i + 1;

}

**Assignment 18: Static Single Assignment Form**

Transform the above code into Static Single Assignment Form (SSA).

i\_0 = 0;

while (i < n\_0) {

j\_0 = 0;

while (j < m\_0) {

if (i\_ < j\_) {

val\_1 = val\_0 + i\_;

} else if (j\_ == i\_) {

val\_2 = val\_0 - 1;

} else { // j > i

val\_3 = val\_0 + j\_;

}

j\_ = j\_ + 1;

}

i\_ = i\_ + 1;

}

**Assignment 19: Machine-Independent Optimisation**

Apply the loop unswitching optimisation to the (original) code above.

**Assignment 20: Compilation Schemes**

Devise a formal compilation scheme that systemativally eliminates all occurences of while-loops

in the body of a CiviC function denition and replaces them by semantically equivalent control

code without while-loops.