Problem 6.1:

```
1       100011 00000 00001 00010 00000000000 # lw r1, 0x1000 (r0) // r1 = input
2       000000 00001 00000 00010 00000100000 # add r2, r1, r0     // r2 = r1
3       001000 00000 00011 0000000000000001  # addi r3, r0, 1     // r3 = 1
4 loop: 000100 00010 00000 0000000000001100  # beqz r2, end       // if r2=0 goto end
5       000000 00010 00011 00011 00000011001 # multu r3, r2, r3   //   r3 = r2 * r3
6       001010 00010 00010 0000000000000001  # subi r2, r2, 1     //   r2 = r2 - 1
7       000010 11111111111111111111110000    # j loop             // goto loop
8 end:  101011 00000 00011 0001000000000100  # sw r3, 0x1004 (r0) // output = r3
```

Dieser Code implementiert die Fakultätsfunktion.
Der Counter r2 wird von dem Wert, der in Register 0x1000 steht, dekrementiert bis er 0 ist.
Bei jedem Schleifendurchlauf wird r2 auf das Endergebnis r3 multipliziert.
$(n \cdot n-1 \cdot ... \cdot 1) = n!$

Problem 6.2:

```
 1 lw r1 , 0 x1000 (r0)     # 10001100000000010001000000000000 ; 0x8c011000
 2 lw r2 , 0 x1004 (r0)     # 10001100000000010000100000000100 ; 0x8c021004
 3 loop : beqz r1 , end     # 00010000001000000000000000010010 ; 0x10200012
 4 slt r3, r1, r2           # 00000000001000100001100000101010 ; 0x0022182a
 5 bnez r3 , branch         # 00010000011000000000000000001110 ; 0x1060000d
 6 sub r3 , r1 , r2         # 00000000001000100001100000100010 ; 0x00221822
 7 add r1 , r2 , r0         # 00000000010000000000100000100000 ; 0x00400820
 8 add r2 , r3 , r0         # 00000000011000000001000000100000 ; 0x00601020
 9 j loop                   # 00001011111111111111111111110000 ; 0x0bffffff0
10 branch: sub r3, r2, r1   # 00000000010000010001100000100010 ; 0x00411802
11 add r2, r1, r0           # 00000000001000000001000000100000 ; 0x00201020
12 add r1, r3, r0           # 00000000011000000000100000100000 ; 0x00600820
13 j loop                   # 00001011111111111111111111110000 ; 0x0bffffff0
14 end: sw 0x1008 (r0), r2  # 10101100000000100001000000001000 ; 0xac021008
```