Your name:
Matric. number:

Computational Physics — A. Scrinzi
Winter term 2016/2017

Exam February 10, 10:00 - 12:00 (120 min)

Resources: This is a strictly paper and pencil exam. No books, other written material, and no electronic devices of any kind are admitted. Exchange of information during the exam will punished by a grade 5 for all involved, donor and acceptor, with no option for repetition of the exam.

Pseudocodes: Some algorithms should be given in the form of commented pseudo-code. A pseudo code is considered only complete, if the algorithm is explained separately verbally and its realization is indicated in the code. An example for pseudo-code is given below.

Return these sheets: An exam is only valid if *all task sheets* are returned.

——— ========== ———

Example for description of an algorithm and pseudo-code

Use the following style for your algorithms and pseudo-codes:

Fast code for Ising model in the Metropolis algorithm
Basic idea: tabulate possible Boltzmann-factors, do not compute total energy, only energy changes.

1. Input: N... number of spins, J... spin coupling, B... magnetic field, kT... temperature (times Boltzmann constant), M... number of steps
2. Initialize spins to $\vec{S} \equiv 1$, energy, seed random generator
3. Get a list of Boltzmann-factors for $E_* - E_m$ when a single s_i is flipped, cases:

   case 0 $s_{i-1}, s_i, s_{i+1} = 1, 1, 1 \to 1, -1, 1 : \exp[(-4J - 2B)/kT]$

   case 1 $s_{i-1}, s_i, s_{i+1} = -1, -1, -1 \to 1, -1, 1 : \exp[(-4J + 2B)/kT]$

   ⋮ ... list all relevant cases, as they would be required in actual code ...

   Peridic boundary conditions: $i \pm 1 \to i \pm 1 \mod N$
4. loop through all steps:
5. randomly select spin: i=r*N, random r in [0,1], i integer
6. de=0, cases 0,1: initial all equal: maximal decrease of energy

7. ... list all relevant steps, as they would be required in actual code ...

Code corresponding to description (here python style, exact python syntax not required)

```
# [1] Input:
def IsingMetropolis(N,J,B,kT,M,printinterval=1):
    """Metropolis algorithm for the Ising model"""

    #[2] seed and initialize
    random.seed(1)
    S=np.ones((N),'int')

    #[3] Boltzmann factors
    expEkT=[exp((-4*J-2*B)/kT),
            exp((-4*J+2*B)/kT),
            exp((    -2*B)/kT),
                  ....          ]
    #[4]
    for n in range(M):
        #[5]
        i=int(N*random.random())

        #[6] different cases...
        if  (S[i]==S[(i+1)%N]) != (S[i]==S[(i-1)%N]): dE=2
        elif S[i]==S[(i+1)%N]: dE=4
        else: dE=0

        if S[i]==-1: dE+=1
        etc.,etc.
```