# Exam: Artificial Intelligence – Algorithms and Application

Module Exam

**Summer 2022** Date: 02.09.2022

## Important Information

TECHNISCHE UNIVERSITÄT DARMSTADT

WIRTSCHAFTS INFORMATIK

- Please check your exam copy for completeness.
  It covers **20 pages** (cover sheet included).
- Fill out the cover sheet immediately after receiving the exam.
- Use only the examination paper to solve the tasks. If you do not have enough space, you can receive additional paper during the examination. Additional papers must also be marked with your name and matriculation number.
- Please leave a **correction margin of 3 cm**.
- You have a total of **90 minutes** to complete the exam.
- Except for a **non-programmable calculator, no other aids** are allowed in the exam.

**We wish you much success!**

**Please fill out clearly in block letters.**

First Name ..... Last Name ..... Seat No. .....

Matr. No. ..... Course of Study .....

❑ Master
❑ Diplom

Repeater:
❑ yes ❑ no

| Section | Max. Points | Achieved Points |
|---|---|---|
| **1** | 34 | |
| **2** | 29 | |
| **3** | 27 | |
| **Sum** | 90 | |

**Exam Review** („Klausureinsicht“):

(do not fill out before the review)

I have reviewed the corrected exam:

☐ There are no complaints about the correction.

☐ Complaints about the correction exist (see additional sheet).

Date: ……..

Signature: ……....

First Name..... Last Name..... Matr. No.....

## 1 Basic Concepts and Algorithms (34 Points)

**1.1** Many people have tried to define the concept "artificial intelligence". The most popular one is from McCarthy. Please give **his definition** of **Artificial Intelligence** that we have discussed in the lecture. (1 P)

**1.2** Imagine you have built a machine learning model that has a **very low bias** and a **very high variance.** Please **state** to which **theoretical problem** these characteristics correspond. (1 P)

**1.3** Please **briefly explain** the **difference** between Python development in the ***Spyder IDE*** and ***Jupyter notebooks***. Please also name **one use case** in which the ***Spyder IDE*** is the **better** choice than ***Jupyter notebooks*** and one use case in which ***Jupyter notebooks*** are the better choice than **Spyder IDE**. (2 P)

First Name..... Last Name..... Matr. No.....

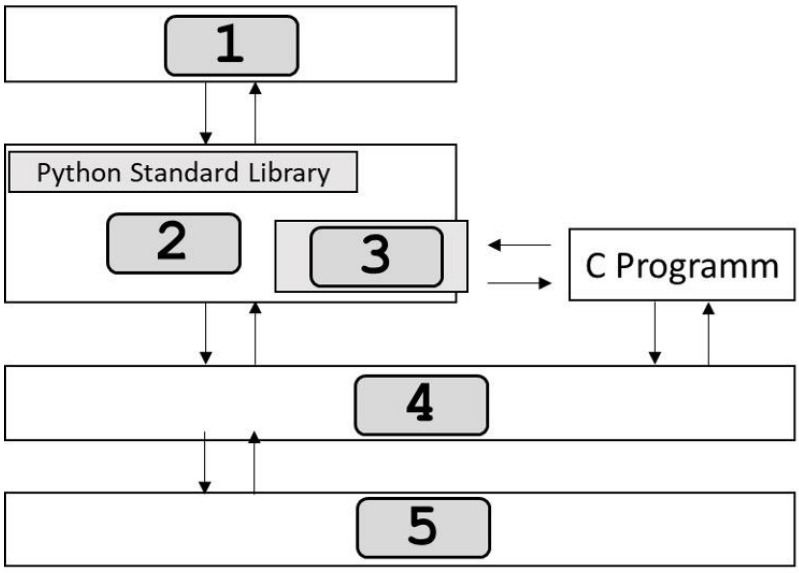
**1.4** Please **briefly explain** what the term ***PCA* stands for** and **what it was used for** in the lecture. (2 P)

**1.5** Is ***feature scaling*** in **general required** after a **normalization** has been applied? Please **briefly explain** your **decision**. (2 P)

**1.6** Please **name three other *scientific domains of AI*** than machine learning and knowledge reasoning that were introduced in the lecture (1.5 P)

First Name..... Last Name..... Matr. No.....

**1.7** In the lecture, we discussed the characteristics and benefits of Python.
Please **insert in the table** the **following *Python concepts*** that are **missing in the figure** to map them to the corresponding numbers: *Hardware, Python Program, Python API, Operating System, Python Interpreter*. (2.5 P)




| Number | Python Concept |
|---|---|
| **1** | |
| **2** | |
| **3** | |
| **4** | |
| **5** | |

First Name..... Last Name..... Matr. No.....

**1.8** Please **briefly explain** the **difference** between a ***route-finding problem*** and the ***touring*** problem. **Which kind of problem** is the ***traveling salesman problem***? (3 P)

**1.9** What **type** of **information system** is "***Mycin***"? (1 P)

**1.10** Please **briefly explain** what the ***conjunctive normal form (CNF)*** constitutes in knowledge reasoning. Please **provide one example** in **CNF**. (2 P)

First Name..... Last Name..... Matr. No.....

**1.11** Please **briefly explain**: What is a "***CAPTCHA***" and **how** is it **related** to **Artificial Intelligence**? (2 P)

**1.12** Please **briefly explain** the following **Python command** we used in the lecture (2 P):

```
pip install -r packages.txt
```

First Name..... Last Name..... Matr. No.....

**1.13** In the lecture, we discussed ***de Finetti's theorem***. Please **summarize** the ***central statements*** of the theorem. Please also construct an "***agent 2***" that **successfully defeats** the following irrational **agent 1**. Please **illustrate** your **explanations** by **computing** the different **actions** and **outcomes**. (8 P)

**Agent 1** has the following **set of degrees of belief**:

$P(A) = 0.3 \quad P(B) = 0.2 \quad P(A \vee B) = 0.9$

First Name..... Last Name..... Matr. No.....

**1.14** The current AI Capstone was conducted in the field of Porsche's complaint management. Please **briefly describe two potentials** of ***Porsche's complaint management*** as a sensor of product and service quality that were presented in the guest lecture by Porsche. (2 P)

**1.15** Please **describe two** possible **data-related limitations** when aiming to create a **deployable AI** solution based on your experience that you gained with the Porsche complaint data during the AI Capstone project. (2 P)

First Name..... Last Name..... Matr. No.....

## 2 Application of Genetic Algorithms (29 Points)

**Consider** the following **AI problem**:

A genetic algorithm uses ***chromosomes*** in the form $c = G_1G_2G_3G_4G_5G_6G_7G_8$.

The chromosomes are of a ***fixed length*** of eight genes $G_i$.

***Each gene*** can be any digit between 0 and 9.

The fitness of an individual *x* is calculated by the following ***fitness function***:

$$fitness = (G_1 + G_2) - (G_3 + G_4) + (G_5 + G_6) - (G_7 + G_8)$$

The ***initial population*** consists of four individuals with the following chromosomes:

$$c_1 = 56415333;\ c_2 = 78176601;\ c_3 = 32291258;\ c_4 = 14854321$$

**2.1** Please **illustrate** the **basic algorithm** of ***genetic algorithms***. Please also **briefly explain** what happens in **each process step**. (5 P)

First Name..... Last Name..... Matr. No.....

**2.2** Please **calculate** the ***fitness*** of **each** of the above given **individuals** and ***rank*** them from the **least fit one** to the **fittest one**. (5 P)

**2.3** Please **perform** the **following crossover operations**. (4 P)

a) Cross the two **least fit individuals** using ***one-point crossover*** in the **middle**.

b) Cross the two **fittest individuals** using a ***two-point crossover*** between $G_3$ and $G_4$ and between $G_5$ and $G_6$.

First Name..... Last Name..... Matr. No.....

**2.4** After multiple evaluations you get the following **population** of **offsprings *O***. Please **evaluate** the ***fitness*** of the **new population** and **explain** if the **fitness** has **improved** or **not**. (5 P)

$o_1$ = 44444444; $o_2$ = 33333333; $o_3$ = 12341234; $o_4$ = 43214321; $o_5$ = 48163248

First Name..... Last Name..... Matr. No.....

**2.5** Please **consider only** the ***fitness function*** and that the ***genes of the distinct chromosomes*** can **only** be ***digits***. Please **find** the ***optimal solution*** that has the **maximum fitness**. Please **explain** your **solution** and your **calculations**. (6 P)

First Name..... Last Name..... Matr. No.....

**2.6** If you **consider** the ***initial* chromosomes**, is a genetic algorithm able to **reach** the **optimal solution from above without** the **mutation** operator? Please explain why or why not. (4 P)

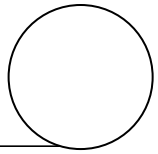
First Name..... Last Name..... Matr. No.....

## 3 Intelligent Agents with Python (27 Points)

The manager of a local construction market hires you to design the agent program for an agent in Python. For this purpose, you have received the following code from your friend Sandra, which she has copied from her lecture "AI Algorithms and Applications with Python".

**Python Code**

```
vacuum_world = {"1":[["2"], False],
                "2":[["1", "3", "4"], False],
                "3":[["2"], True],
                "4":[["2"], False]}

class Cleaner:
    def __init__(self, room, world):
        self.location = room
        self.world = world

    def percepts(self):
        is_dirty = self.world[self.location][1]
        self.act(is_dirty)

    def drive(self):
        neigbor_rooms = self.world[self.location][0]
        num_rooms = len(neigbor_rooms)
        r = numpy.random.randint(low = 0, high = num_rooms)
        self.location = neigbor_rooms[r]

    def suck(self):
        self.world[self.location][1] = False

    def act(self, ______ ):
        if(is_dirty == True):
            self.suck()
        else:
            self.drive()
```



First Name..... Last Name..... Matr. No.....

**3.1** Based on Sandra's code on the prior page, please **classify** the ***type of agent*** Sandra has implemented and **briefly explain** your **decision.** (2 P)

**3.2** Sandra uses a specific Python data structure to **model** the **vacuum_world**. **Which kind of *data structure*** does she use to do so? Please also **draw** a **map** of the **vacuum_world** based on the information from the code and **mark** the **agent's starting position** in the map. (4 P)

First Name..... Last Name..... Matr. No.....

**3.3** You try to run the code and you receive the following error message:

```
NameError: name 'numpy' is not defined
```

**How** do you **fix** this **error**, and **what** is "***numpy***"? (2 P)

**3.4** Moreover, the `act()` function is missing a parameter: `def act(self, _____ )`

Please **correct** the **function** and **fill** in the **missing code**. (2 P)

First Name..... Last Name..... Matr. No.....

**3.5** The current agent version is missing an energy management. Please **extend** the **agent class** with a **"power consumption" functionality** in Python considering the following rules (9 P):

- The agent consumes one energy level every time it sucks up some dirt.
- If the energy level drops below 5, the agent should drive back to the location "1".
- The starting energy is 10 energy units.

**Python Code**

```
class Cleaner:



    def __init__(self, room, world):
        self.location = room
        self.world = world




    def percepts(self):
        is_dirty = self.world[self.location][1]
        self.act(is_dirty)




    def drive(self):
        neigbor_rooms = self.world[self.location][0]
        num_rooms = len(neigbor_rooms)
        r = numpy.random.randint(low = 0, high = num_rooms)
        self.location = neigbor_rooms[r]
```

First Name..... Last Name..... Matr. No.....

***Note**: This code block continues on the next page.*

**Python Code**

```
    def suck(self):
        self.world[self.location][1] = False

    def act(self, _______ ):
        if(is_dirty == True):
            self.suck()
        else:
            self.drive()
```

First Name..... Last Name..... Matr. No.....

**3.6** Please **write** some **Python code** to **start** a **cleaning simulation** with the **agent.** (6 P)

First Name..... Last Name..... Matr. No.....

**3.7** What will be a **possible problem** of this **type of agent** if the **number of rooms** in the construction market ***increases***? **How** can you **solve** it? (2 P)