BABEŞ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Bachelor Degree Written Exam 2026
Artificial Intelligence in English

VARIANT 1

REMARKS

- All subjects are compulsory and full solutions are requested.
- The minimum passing grade is 5.00.
- The working time is 3 hours.

SUBJECT Algorithms and Programming

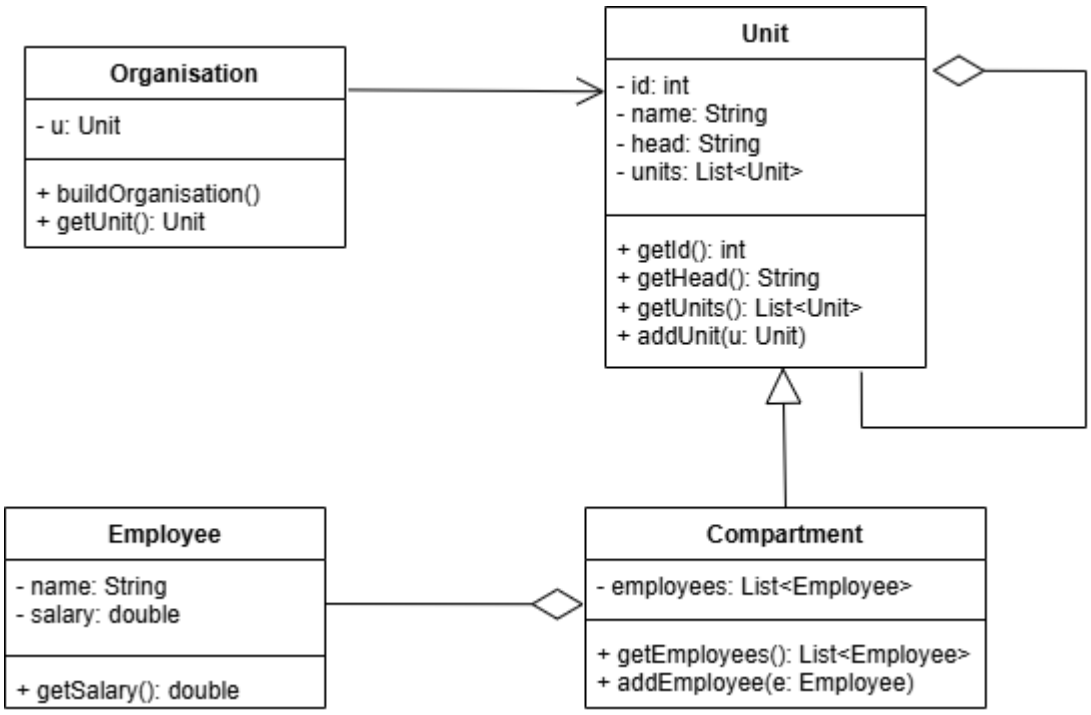
- **The programming language that is used must be indicated.**
- **The implementations are not required to deallocate dynamically allocated memory areas.**
- **Lack of appropriate programming style (suggestive variable names, indentation of code, comments, if necessary, readability of code) will result in a 10% deduction from the subject's score.**
- **Do not add additional attributes or methods, other than those mentioned in the statement, except for constructors and destructors, if applicable. Do not change the visibility of attributes specified in the statement.**
- **Do not use sorted containers, predefined sort or search operations.**
- **Existing libraries (from C++, Java, C#, Python) may be used for data types.**

1. **(3 points)** Write a function in one of the programming languages **Python, C++, Java** or **C#**, which receives as parameters an array *v* containing real numbers sorted in descending order (representing the salaries of employees in a compartment) and a real number *salary* (representing the salary of a new employee). The function will insert *salary* into the array *v* so that the array remains sorted in descending order of salaries. Use an auxiliary function, with an iterative implementation, which returns, in $O(\log_2 n)$ time complexity, the position at which the *salary* should be inserted into the array *v*, where *n* is the length of the array *v*. *Recursive solutions will be partially scored.*
   Note. Inserting an element at position *k* into the array *v* involves shifting the elements at positions *k, k+1, ..., n* one position to the right, after which the length of the array increases by 1. ***Do not use predefined insertion functions from libraries to insert an element at a specific position within the array.***

2. **(1 point)** Indicate the *best-case*, *average-case*, and *worst-case* time complexity for the function from item **1**. Justify your answer.

3. **(5 points)** Consider the following UML diagram, which contains the classes **Organisation** (represents the hierarchical structure of an organization), **Unit** (Organisational unit), **Compartment**, **Employee**.

Note ***The class constructors are not shown in the diagram.***



- The class **Organisation** represents an organization (represented as a hierarchical structure of organizational units). The **getUnit()** method returns attribute *u* of the class.
  - The method **buildOrganisation**() creates the hierarchical structure of the organisation. *Implementation of this method is not required.*
- The class **Unit** represents the hierarchical structure of an organizational unit. It has an identifier (*id*), a name (*name*), and a director (whose name is stored in the attribute *head*). The unit has several (sub)units under its direct authority (stored in the attribute *units*). The class has methods for accessing *id*, *head* and *units* and a method **addUnit(*u*: Unit)** that adds unit *u* to the end of the *units* list.
- The class **Compartment** represents a unit that has no other (sub)units under its direct authority. It stores a list of employees (**Employee**) in the *employees* attribute. The list *employees* stores the employees in descending order of their salaries. The class has a method **addEmployee**(*e*: **Employee**) that adds an employee to the sorted *employees* list and a method **getEmployees()** that returns the list of employees in the compartment.
- The class **Employee** represents an employee identified by a *name* and a *salary*. The class has a method **getSalary()** that returns the *salary* attribute.

Note: *The units (***Unit***) within the organization (***Organisation***) have distinct identifiers (id).*

Write a program that implements the following requirements, using one of the **C++, Java**, or **C#** programming languages:

a) Declare all classes, attributes and methods as per the diagram above. Implement only the constructor of the class **Compartment**.

b) Define a function that receives as parameter an object *o* of type **Organisation** and an integer number *n* and returns a list containing all employees under the authority (direct or indirect) of the organisation's unit having the *id* equal to *n*. The returned list will be empty if the organization does not have a unit with an *id* equal to *n*.

c) Define a function that receives as parameter an object *o* of type **Organisation** and returns a list of compartment heads within the organisation *o*.

d) Create an object *o* of type **Organisation**, call the method **buildOrganisation()**,then call the functions from **b)** and **c)** for the created organisation.

VARIANT 1

SUBJECT Metaheuristics (3 points)

A store needs workers for 2 shifts every day: morning and afternoon. The store has **n** employees available, each having a cost per shift. The rules for the shift scheduling are: (i) each shift must have exactly one employee, (ii) each employee can work maximum 1 shift in a day, and (iii) the total cost should be minimized.

For each of the following requirements, provide a clear solution and explain it.

a) Identify a binary representation for a solution to the problem. Define and justify the size of the resulting search space.

b) Define a neighbourhood function to determine neighbouring solutions and an evaluation function to determine the quality of a solution under the selected binary representation. Explain how you deal with the constraints.

c) Define a crossover operator that can be used for this problem under the binary representation in an evolutionary algorithm. Give an example of its application for n=3 and two parents randomly selected.

VARIANT 1

SUBJECT Machine Learning (3 points)

Detecting Faulty Bikes in a Rental System: in a busy city, a bike-sharing company operates hundreds of bicycles distributed across multiple stations. Customers rely on these bikes for daily commuting, so ensuring that each bike is in good condition is essential. Each time a bike is returned, the system records three numerical measurements collected automatically: average vibration level during the last ride, brake response time, pedaling efficiency score. Occasionally, technicians inspect bikes manually and check if the bike in good condition or it is a faulty bike (needs maintenance). So far, only 120 bikes have been inspected and labeled. Real-world riding conditions introduce a lot of uncertainty: rough roads, rider behavior, and weather conditions create strong noise in the recorded measurements, a perfectly fine bike might show high vibration if used on cobblestone streets, a faulty bike might appear normal if ridden gently. As a result, the data is messy and overlapping, making it difficult to distinguish good bikes from faulty ones. The company wants to build a classification model that can take the above numerical features from a bike and predict whether it is faulty or in good condition. It prepares a training setup for an artificial neural network with: five hidden layers with 50 ReLU neurons/layer, one sigmoid output neuron and an optimization algorithm with the learning rate 0.001 during 500 epochs. The trained model shows a training accuracy of 99% and a validation accuracy of 63%.

Tasks:

a) Draw the architecture of the network as a graph.
b) Compute the total number of trainable parameters.
c) Estimate the number of multiplications in a forward pass.
d) Propose a regularization technique to reduce overfitting and explain why they would help for this architecture.
e) Identify an architectural change that could reduce overfitting and justify it.

VARIANT 1

SUBJECT Graphs (3 points)

A factory has a machine that they want to use to manufacture one product. The manufacturing process has multiple steps, where each specific step can be started possibly only *after* some other specific steps were finished (e.g. step 3 can be started only after step 2 finished; or step 5 can be started after steps 4 and 1 finished). We need to find the correct order of steps so that all the conditions are met.

For each of the following requirements, provide a clear solution and explain it.

a) Describe how you would model this problem as a graph and what type of graph you would use.

b) Given a manufacturing process with 7 steps with the following constraints:
   - steps 2 and 3 can be started only after step 1
   - steps 2 and 5 can be started only after step 4
   - step 6 can be started only after steps 2, 3 and 5
   - step 7 (the final step) can be started only after step 5 and 6

   Draw the graph that represents this particular process.

c) How do you find the correct order of the steps from such a graph? Describe an algorithm that solves the problem and show how it works in the previous example.

BAREM INTELIGENȚĂ ARTIFICIALĂ
VARIANTA 1

Subiect Algoritmică și Programare

Oficiu – **1p**

Cerința **1. – 3p**

Funcția de bază – **0.4p** din care
- signatura – 0.1p
- implementare – 0.3p

Funcția auxiliară – **2.6p** din care
- signatura – 0.1p
- implementare iterativă funcție auxiliară având complexitate timp $O(\log_2 n)$ – 2.5p
  * soluție recursivă având complexitate timp $O(\log_2 n)$ – 1p
  * soluție iterativă având complexitate timp $O(n)$ – 0.5p

Cerința **2. – 1p**
- caz favorabil **0.3p** din care
  - complexitate – 0.15
  - justificare – 0.15
- caz mediu **0.4p** din care
  - complexitate – 0.2
  - justificare – 0.2
- caz defavorabil **0.3p** din care
  - complexitate – 0.15
  - justificare – 0.15

Cerința **3.a) – 1.7p**

Definirea clasei Organisation – **0.2p** din care
- atribut – 0.1
- metode **buildOrganisation**, **getUnit** – 0.1

Definirea clasei Unit – **0.5p** din care
- atribute – 0.2
- constructor – 0.1
- metode **getId, getHead, getUnits, addUnit** – 0.2

Definirea clasei Compartment– **0.7p** din care
- relația de moștenire – 0.2
- atribut – 0.1
- **constructor (a1)** – 0.3
- metode **getEmployees, addEmployee** – 0.1

Definirea clasei Employee – **0.3p** din care
- atribute– 0.2
- metoda **getSalary** – 0.1

Funcția **3.b)** – **2p**
- signatura – **0.1p**
- implementare funcție – **1.8p**
- returnare rezultat – **0.1p**

Funcția **3.c)** – **1p**
- signatura – **0.1p**
- implementare funcție – **0.8p**
- returnare rezultat – **0.1p**

Funcția principală **3.d)** – **0.3p**
- construire obiect **o** – 0.1p
- apel metoda **buildOrganisation**() – 0.1p
- apel funcții b) și c) – 0.1p

BAREM INTELIGENȚĂ ARTIFICIALĂ
VARIANTA 1

Subiect Inteligență Artificială

Oficiu – 1p

Subiect Metaeuristici – 3p

Cerința a) – 1p
- Definire reprezentare binară – 0.5 p
- Definire şi justificare mărime spaţiu de căutare – 0.5p

Cerinta b) – 1p
- Definire funcţie de evaluare – 0.5p
- Definire funcţie de vecinătate – 0.25p
- Explicare şi tratare corectă a restricţiilor – 0.25p

Cerința c) – 1p
- Definire operator încrucişare – 0.75p
- Exemplu de aplicare a operatorului de încrucişare – 0.25p

Subiect Machine Learning – 3p

Cerința a) un graf cu 6 straturi: cu 3, 50, 50, 50, 50, 50, 1 neuroni – 0.5p

Cerința b) nr de param = (3 + 1) x 50 + 4 x [(50 + 1) x 50] + (50 + 1) x 1 = 10 451 – 0.5p

Cerința c) nr de inmultiri 3 x 50 + 4 x (50 x 50) + 50 x 1 = 10 200 – 0.5p

Cerința d) – 0.75p
- descrierea unei tehnici de regularizare – 0.5p
- o scurta justificare – 0.25p

Cerința e) – 0.75p
- descrierea unei tehnici de reducere – 0.5p
- o scurta justificare – 0.25p

Subiect Grafe – 3p

Cerința a) – 1p
- Menționează că trebuie graf orientat + justificare – 0.5p
- Menționează că trebui să fie fără cicluri (i.e DAG) + justificare – 0.5p

Cerința b) Desenează corect graful – 0.5p

Cerința c) – 1.5p
- Determină că o sortare topologică rezolvă problema – 0.25p
- Descrie un algoritm corect de sortare topologică – 0.75p
- Demonstrează corect algoritmul pe graful dat – 0.5p