**SUBIECTE POO**

## SUB 1

3p) Define classes that allow the management of data related to a **weather condition** (date, temperature, description, location, etc.), using members of public, private, protected, const, static type. The class contains at least one dynamically allocated field (like recorded temperatures array), constructors, 2 accessor methods (set will validate the received value) for an attribute of your choice and the class destructor.

(1p) Overload the ++ operator with the postfix form that allows you to change the schedule by delaying it by one hour

(1p) Exemplify the overloading concept for the previously defined set method.

(1p) Using the previous class,exemplify the three different situations in which the copy constructor is called.

(2p) To exemplify the use of a “has a” type 1: M relation (one to several) by defining an additional class to manage the weather conditions for a region with multiple locations. You can use a STL collection but it is not mandatory.

(1p) Overload an operator of your choice to add new weather conditions data for the region.

(1p) Explain how virtualization works. You can use the debugger for that.

## SUB 2

(3p) It is considered an application used to manage the team structure based on the **team members** roles, etc. Specific attributes will be covered, such as: team name, coordinator name, number of team members, members names list, etc. Member data is private or protected. The class contains at least three fields,of which one is dynamically allocated (e.g. names list as an array of strings), a parameter constructor, destructor, accessory methods (get and set) for one of the attributes. Use a constant member . The set () method validates the input data.

(1p) Overload the operator - =, to change the value of an attribute by multiplication.

(1p) Overload the function operator, in the form (string value, int index) to verify if the received value is the name of a team member on the index position.

(2p) Exemplify the concept of an "is a" relation by defining a class that manages an upgrade of the previous configuration. Test the solution in the main ().

(2p) Exemplify the concept of working in C ++ with binary files (read and write) to save and restore the value of an object (for one of the defined classes).

(1p) Explain and exemplify the polymorphism concept in C++.

The ex officio/bonus/free point is included in the first requirement. Failure to implement any of the requirements will result in the exam being graded with 1. In order to be considered, the solutions must not contain compilation errors.The implementation of the solution must be accompanied by a description of the concepts used.All requirements must be tested in main()

## SUB 3

(3p) Define classes that allow the abstraction of the **video conferencing concept** (name, platform used, start time, etc.) Member data is private or protected. The class contains at least three fields, of which one is dynamically allocated, a parameter constructor, destructor, accessory methods (get and set) for one of the attributes.Use a constant member.The set() method validates the input data.

(1p) Overload the + operator in the form value + object to change the value of an attribute.

(1p) Overload at least one relational operator to be able to compare two video conferences based on start date

(2p) Exemplify the concept of a "has a" relation by defining a new class that handles several objects of the previous class type.Implement a method that allows you to add an object to your collection.You can use STL collections but it is not mandatory.Test the solution in the main().

(2p) Exemplify the concept of serialization and deserialization of an object by writing its values in a binary file.

(1p) Explain and exemplify the concept of "dangling pointer" using a pointer to one of the previously defined classes.

## SUB 4

(3p) Define classes that allow the management of data related to a **printer**(manufacturer, model, number of printed pages, etc.), using members of public, private, protected, const, static type. The class contains at least one dynamically allocated field (like model as an array of chars), constructors, 2 accessor methods(set will validate the received value) for an attribute of your choice and the class destructor.

(1p) Overload the cast operator to real value in the explicit form that allows the temperature to be returned in fahrenheit degrees((0° C × 9 / 5) + 32).

(1p) Exemplify the concept of transferring objects by reference to a method.

(1p) Exemplify the concept of vector of pointers to objects(definition and initialization).Go through the vectorand call a class method.

(2p) Exemplify the concept of "is a" type relation by specializing the defined classes.The subclass constructors(defaultand with arguments) will explicitly call the base class one.Test the solution by instantiating the new class.

(1p) Exemplify the concept of late - binding and virtual methods.

(1p) Exemplify the concept of template method / class in C++.Call it in main().

## SUB 5

(3p) It is considered an application used to manage the **characters in a computer** / mobile game.Specific attributes will be followed, such as: character name, life points, special powers, whether he is a positive or negative hero, etc. Member data is private or protected.The class contains at least three fields, of which one is dynamically allocated, a parameter constructor, destructor, accessory methods (get and set) for one of the attributes. Use a static or const.The set() method validates the input data.

(1p) Overload the += operator in the form object += value to change the value of an attribute.

1p) Define the >= operator that compares two objects and returns true based on the value of an attribute.

2p) Exemplify the concept of "is a" type relationship by specializing the previous class.The new class adds a new attribute and the parameter constructor explicitly calls the base class constructor.Test the solution by instantiating the new class.

(2p) Explain and exemplify the concept of abstract class.Derive one of the existing classes from the abstract class and test in the main().

(1p) Explain and exemplify the concept of "dangling pointer" using a pointer to one of the previously defined classes.

## SUB 6

(3p) Define a class that allows the management of data related to an **exam subject** (title, points for each requirement, exam date, requirements, etc.), using members of public, private, protected, const, static type. The class contains at least one dynamically allocated field, constructors, 2 accessor methods (set will validate the received value) for an attribute of your choice and the destructor.

(1p) Overload the + operator to allow the implementation of the string + object operation.

(2p) Overload the << operator to allow the object to be written to a text file. Test in main () on a locally created text file.

(2p) Exemplify the use of a 1: 1 type "has a" relationship (one to one) by defining an additional class to manage a student's assessment. For the new class define the copy constructor.

(1p) Exemplify the relationship between the constructors of the two classes, which are in the ratio of 1: 1, if the initial class does not have a default constructor.

(1p) Explain and exemplify the concept of "dangling pointer" using a pointer to one of the previously defined classes.

## SUB 7

(3p) It is considered an application used to manage the **configuration of a laptop**. Specific attributes will be followed, such as: model, processor type, video card, component list, warranty, housing series, etc. Member data is private or protected. The class contains at least three fields, of which one is dynamically allocated, a parameter constructor, destructor, accessory methods (get and set) for one of the attributes. Use a constant member. The set () method validates the input data.

(1p) Overload the operator * in the form object * value to change the value of an attribute by multiplication.

(1p) Overload the cast to string operator, in the explicit form to return the description of an object. (...)

## SUB 8

(3p) It is considered an application used to manage the route between **tourist points**. Specific attributes will be defined, such as: distance, tourist points, duration, points of interest, map, etc. Member data is private and access methods are provided. The class contains at least three fields,of which one is dynamically allocated, a parameter constructor, destructor, accessor methods (get and set) for one of the attributes. The set () method validates the input data. Use a static or const.

(1p) Overload the operator - (minus) for object- value to change the value of an attribute.

(1p) Define the == operator that compares two objects and returns true if all attribute values are equal to each other.

(2p) Exemplify the concept oftype relationship “is a” by specializing the previous class. The new class adds a new attribute and the parameter constructor explicitly calls the base class constructor. Test the solution by instantiating the new class.

(2p) Explain and exemplify the concept of late-binding by defining a virtual method in the base class.

(1p) Exemplify the concept of template function in C ++.

## SUB 9

(2p) Se consideră aplicația pentru gestiunea documentelor dintr - un **dosar de admitere** la facultate, folosind aspecte comune precum foaie matricolă, copie CI, copie diplomă certificat naștere, opțiuni facultăți, etc.Definiți o clasă care modelează un aspect propriu acestei activități.Datele membre sunt private și sunt puse la dispoziție metode de acces.Clasa conține cel puțin patru câmpuri, dintre care unul este alocat dinamic, constructori, metodele specifice claselor cu membri alocați dinamic.Folosiți un membru static sau const.

(1p) Supraîncărcați operator+ pentru adăugarea unei noi opțiuni în cadrul dosarului de admitere.

(2p) Să se genereze un raport într - un fișier text cu informațiile dintr - un dosar de admitere la facultate.

(2p) Să se exemplifice utilizarea mecanismul try - catch pentru a gestiona modificarea valorii unui atribut pentru care a fost implementată metoda setter.Metoda setter trebuie sa genereze o excepție specifica aplicației(definita de programator) pentru date invalide.Utilizatorul este avertizat dacă introduce date invalide de la consolă având posibilitatea sa le reintroducă.

(1p) Definiți și testați o metodă statică pentru afișarea numărului total de dosare create la un moment dat.Metoda va contoriza dosarele create chiar dacă nu au fost depuse la facultate.

(2p) Să se exemplifice conceptul de relație de “has a” între clase prin crearea unei clase pentru reprezentarea unui student care depune un dosar de admitere la facultate.

## SUB 10

(3p) Se consideră o aplicație pentru **gestionarea caselor dintr - un sat**.Se vor urmări atribute specifice, precum: suprafață, numar camere, cu sau fără etaj, etc.Datele membre sunt private și sunt puse la dispoziție metode de acces.Clasa conține cel puțin patru câmpuri, dintre care unul este alocat dinamic, constructori, metodele specifice claselor cu membri alocați dinamic și operatorul de afișare.Folosiți un membru static sau const.

(1p) Se definește operator() (int) (operator funcție) care primește numărul de locatari din acea casă.Operatorul returnează true dacă fiecare persoană are posibilitatea să aibă propria camera în casă sau false daca sunt mai puțin camera decât personae.

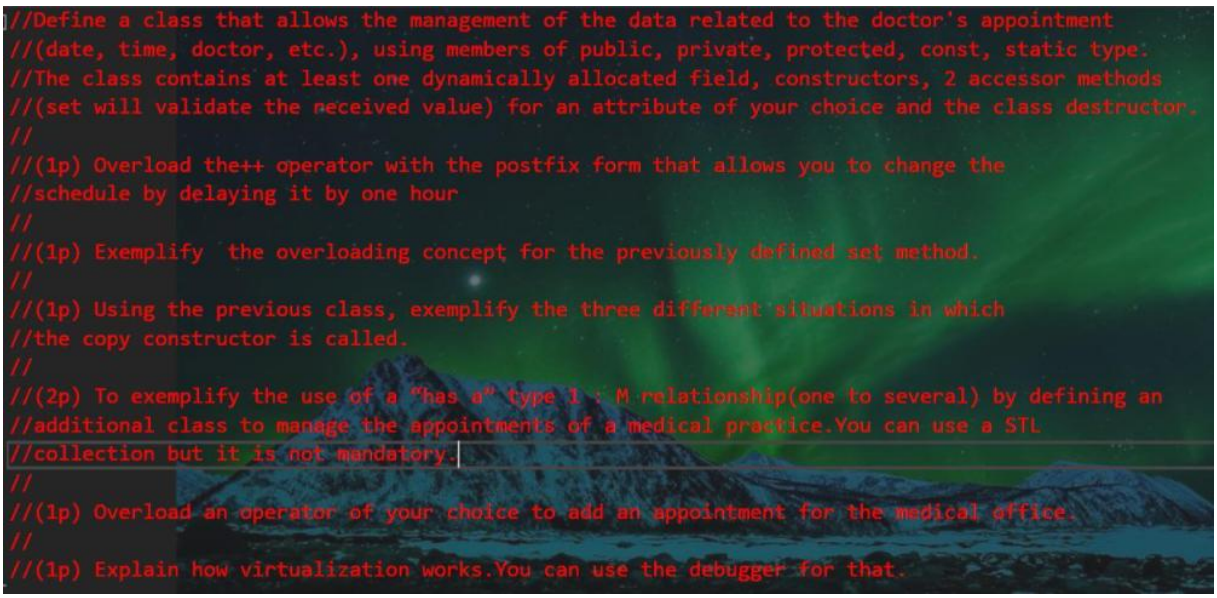
(1p) Definiți operatorul = = care compară două obiecte de tip Casa și returnează true dacă toate valorile atributelor sunt egale între ele.

(2p) Exemplificați conceptul de relație de tip „is a” prin specializarea clasei Casa.Testați soluția prin instanțierea noii clase.

(2p) Explicați conceptele de early binding și late binding.

(1p) Exemplificați conceptul de funcție template în C++.

# SUB 11


//Define a class that allows the management of the data related to the doctor's appointment
///(date, time, doctor, etc.), using members of public, private, protected, const, static type:
//The class contains at least one dynamically allocated field, constructors, 2 accessor methods
///(set will validate the received value) for an attribute of your choice and the class destructor.
//
///(1p) Overload the++ operator with the postfix form that allows you to change the
//schedule by delaying it by one hour
//
///(1p) Exemplify  the overloading concept for the previously defined set method.
//
///(1p) Using the previous class, exemplify the three different situations in which
//the copy constructor is called.
//
///(2p) To exemplify the use of a "has a" type 1 : M relationship(one to several) by defining an
//additional class to manage the appointments of a medical practice.You can use a STL
//collection but it is not mandatory.
//
///(1p) Overload an operator of your choice to add an appointment for the medical office.
//
///(1p) Explain how virtualization works.You can use the debugger for that.