**Tema 1**

1. **cmake build -3p**
2. **incapsulare -1p**
3. **initializare 1p**
4. **dealocare memorie din heap in destructor**
5. **copy constructor & assignment operator 2p**

**Tema 2**

1. **cmake build -3p**
2. **mostenire – 1p**
3. **abstractizare – 1p**
4. **folosire namespace -> creare propriu namespace -1p**
5. **move constructor (sa punem mesaj sa vedem in ce context e apelat) - 2p**
6. **implementam o clasa abstracta (= interfata abstracta) are metode virtuale**
   * **metoda virtuala pura inseamna ca musai trb sa fie implementata**

**Tema 3**

1. **3p cmake 3 directoare**
2. **1p folosim colectii (vector, list) = clase templetizate ; de ce am ales; ce e mai eficient**
3. **1p facem o functie templetizata si sa o folosim cum trebuie deci cu mai multe tipuri de date**
4. **1p clasa templetizata**
5. **2p folosim unique pointer; de ce am avut nevoie de el**

*(smart pointer - obiect care are la baza un pointer normal care stie singur sa manage uiasca singur memoria*

*unique pointer = smart pointer - nu putem sa il copiem; putem doar sa il mutam - nu e recomandat)*

1. **2p share pointer - inversul lui unique;**

*putem sa il copiem si sa il share-uim*

*in constructor incrementam variabila statica si in destructor o decrementam pentru SHARED pointer, resursa se va elibera cand variabila ajunge pe 0 si atunci se eliberaza memoria*