

## FIȘA DISCIPLINEI

*Fundamentele programării*

Anul universitar 2026 - 2027

### 1. Date despre program

1.1. Instituția de învățământ superior	<b>Universitatea Babeș-Bolyai din Cluj-Napoca</b>
1.2. Facultatea	<b>Facultatea de Matematică și Informatică</b>
1.3. Departamentul	<b>Departamentul de Informatică</b>
1.4. Domeniul de studii	<b>Informatică</b>
1.5. Ciclul de studii	<b>Licență</b>
1.6. Programul de studii / Calificarea	<b>Informatică (în limba engleză)</b>
1.7. Forma de învățământ	<b>cu frecvență</b>

### 2. Date despre disciplină

2.1. Denumirea disciplinei	<b>Fundamentele programării</b>			Codul disciplinei	<b>MLE5005</b>
2.2. Titularul activităților de curs	<b>Assoc. Prof. PhD. Molnar Arthur</b>				
2.3. Titularul activităților de seminar	<b>Assoc. Prof. PhD. Molnar Arthur</b>				
2.4. Anul de studiu	<b>1</b>	2.5. Semestrul	<b>1</b>	2.6. Tipul de evaluare	<b>Examen</b>
2.7. Regimul disciplinei	<b>Obligativu</b>		2.8. Tipul disciplinei	<b>Disciplină de specializare (DS)</b>	

### 3. Timpul total estimat (ore pe semestru al activităților didactice)

3.1. Număr de ore pe săptămână	<b>6</b>	din care: 3.2. curs	<b>2</b>	3.3. seminar/ laborator/ proiect	<b>4</b>
3.4. Total ore din planul de învățământ	<b>84</b>	din care: 3.5. curs	<b>28</b>	3.6 seminar/laborator	<b>56</b>
<b>Distribuția fondului de timp pentru studiul individual (SI) și activități de autoinstruire (AI)</b>					<b>ore</b>
Studiul după manual, suport de curs, bibliografie și notițe (AI)					15
Documentare suplimentară în bibliotecă, pe platformele electronice de specialitate și pe teren					20
Pregătire seminare/ laboratoare/ proiecte, teme, referate, portofolii și eseuri					20
Tutoriat (consiliere profesională)					5
Examinări					0
Alte activități					6
<b>3.7. Total ore studiu individual (SI) și activități de autoinstruire (AI)</b>				<b>66</b>	
<b>3.8. Total ore pe semestru</b>				<b>150</b>	
<b>3.9. Numărul de credite</b>				<b>6</b>	

### 4. Precondiții (acolo unde este cazul)

4.1. de curriculum	-
4.2. de competențe	-

### 5. Condiții (acolo unde este cazul)

5.1. de desfășurare a cursului	Sală de curs cu video-proiector și acces la Internet
5.2. de desfășurare a seminarului/ laboratorului	Sală de seminar cu video-proiector și acces la Internet

### 6.1. Competențele dobândite în urma absolvirii programului de studii (se preiau din planul de învățământ)<sup>1</sup>

<sup>1</sup> Se vor prelua din Planul de învățământ al programului de studii acele competențe profesionale și/sau transversale la dezvoltarea cărora contribuie disciplina pentru care se elaborează fișa disciplinei. Pentru fiecare competență se va prelua întregul enunț, inclusiv codul competenței, cu formularea care apare în planul de învățământ, fără modificări. Dacă nu se preia nici o competență din oricare din cele două categorii, se șterge linia din tabel aferentă acelei categorii.

Competențe profesionale	
Codul competenței	Competență
CP1	crează softuri
CP7	proiectează sistemul informatic
CP9	remediază erorile din software
CP11	utilizează șabloane de proiectare de software
Competențe transversale	
Codul competenței	Competență
CT1	Lucrează independent
CT3	Gândește analitic

## 6.2. Rezultatele învățării specifice programului de studii (se preiau din planul de învățământ)<sup>2</sup>

Rezultatele învățării vizate prin disciplină		
Codul competenței	Cunoștințe și înțelegere (Knowledge and understanding)	Abilități academice specifice (Specific academic skills)
CP4	Studentul/absolventul cunoaște și explică paradigmele moderne de programare, arhitecturi software și metodologii de dezvoltarea proiectelor software.	Studentul/absolventul proiectează, planifică, construiește, dezvoltă aplicații software scalabile și utilizează eficient resursele hardware și software.

## 7. Rezultatele învățării specifice disciplinei

Cunoștințe și înțelegere (Knowledge and understanding)
Să cunoască conceptele de bază ale ingineriei software (proiectare, implementare și mentenanță) și să fie capabil de a le aplica cu scopul de a crea aplicații de dimensiune mică și medie în limbajul Python folosind paradigma programării structurate sau orientate obiect și care utilizează interfața cu utilizatorul în consolă.
Abilități academice specifice (Specific academic skills)
Să cunoască conceptele cheie ale programării.
Să cunoască conceptele de bază ale ingineriei software în domeniul proiectării, implementării și mentenanței sistemelor software.
Să capete o bună înțelegere asupra uneltelor software de bază utilizate în procesul de dezvoltare și testare.
Să cunoască limbajul Python și să dobândească competențele necesare executării, testării și depanării programelor Python.
Să dobândească și să își îmbunătățească propriul stil de programare.

## 8. Conținuturi

8.1 Curs	Metode de predare - învățare	Observații <sup>3</sup>
<b>Introducere în curs</b> Ce este programarea: algoritm, program, elementele de bază ale limbajului Python, interpretorul Python.	<ul style="list-style-type: none"> <li>Expunere interactivă</li> </ul>	

<sup>2</sup> Se menționează rezultatele învățării specifice programului de studiu la dezvoltarea cărora contribuie disciplina pentru care se elaborează fișa. Enunțurile, preluate fără modificări din Planul de învățământ în funcție de tipul disciplinei (DF/DS/DC) se trec în dreptul competenței asociate.

<sup>3</sup> De exemplu aspecte organizatorice, recomandări pentru studenți, aspecte specifice legate de curs/seminar cum ar fi invitarea unor practicieni în domeniu etc.

<p><b>Recursivitatea. Complexitatea algoritmilor</b></p> <ul style="list-style-type: none"> <li>• Noțiunea de recursivitate, recursivitate directă și indirectă, exemple.</li> <li>• Analiză empirică și asimptotică .</li> <li>• Notăția asimptotică: big-o, little-o, big-omega, little-omega, theta.</li> <li>• Compararea algoritmilor din punct de vedere al eficienței, analizarea atât a complexității timp cât și a complexității spațiului de memorie.</li> </ul>	<ul style="list-style-type: none"> <li>• Explicație</li> <li>• Conversație</li> <li>• Exemple</li> <li>• Demonstrație didactică</li> </ul>	
<p><b>Căutare. Sortare</b></p> <ul style="list-style-type: none"> <li>• Specificarea problemelor de căutare și sortare.</li> <li>• Metode de căutare: secvențială, binară, exponențială.</li> <li>• Metode de sortare: a bulelor, prin selecție, prin inserție (și optimizări), quick sort, prin interclasare (și optimizări).</li> <li>• Complexitatea timp și a spațiului de memorie pentru algoritmi de căutare și sortare.</li> </ul>		
<p><b>Metode de rezolvare a problemelor</b></p> <ul style="list-style-type: none"> <li>• Prezentarea metodelor greedy, divide et impera, backtracking și a programării dinamice.</li> <li>• Probleme tipice acestor metode.</li> <li>• Analizarea complexității timp și a spațiului de memorie.</li> </ul>		
<p><b>Programarea procedurală</b></p> <ul style="list-style-type: none"> <li>• Tipuri compuse în Python: listă , tuplă , dicționar.</li> <li>• Funcții: definiție, crearea cazurilor de testare, domeniu de vizibilitate al variabilelor, apel, transmiterea parametrilor.</li> <li>• Pașii metodologiei de dezvoltare bazate pe teste și refactorizarea.</li> </ul>		
<p><b>Programarea modulară</b></p> <ul style="list-style-type: none"> <li>• Ce este un modul, ce este un modul în Python, domeniul de vizibilitate al variabilelor într-un modul, pachete, biblioteci standard de module, instalarea modulelor adiționale.</li> <li>• Organizarea codului sursă : responsabilități, principiului singurei responsabilități, separarea responsabilităților, dependența, cuplarea, coeziunea.</li> <li>• Introducere în șablonul arhitecturii stratificate.</li> </ul>		
<p><b>Clase și obiecte</b></p> <ul style="list-style-type: none"> <li>• Definirea tipurilor de date adiționale în Python prin utilizarea claselor.</li> <li>• Metodele claselor, metode speciale, crearea obiectelor, vizibilitatea claselor și a pachetelor Python.</li> </ul>		
<p><b>Programarea bazată pe obiecte</b></p> <ul style="list-style-type: none"> <li>• Ascunderea informației, încapsularea, moștenirea, tipizarea dinamică.</li> <li>• Aplicarea principiilor arhitecturii stratificate folosind clase și obiecte.</li> <li>• Utilizarea mecanismului de excepții.</li> <li>• Exemple focalizate pe soluții bazate pe arhitectura stratificată ce demonstrează utilizarea ierarhiilor de clase, a fișierelor text și binare în scopul stocării datelor precum și utilizarea excepțiilor Python pentru a semnală apariția unei situații neașteptate.</li> </ul>		
<p><b>Principii pentru proiectarea programelor</b></p> <ul style="list-style-type: none"> <li>• Principii de proiectare: expertul în informație, cuplarea scăzută, coeziunea înaltă, principiul singurei responsabilități, injectarea dependențelor.</li> <li>• Limbajul Unified Modelling Language, diagramele de clase UML.</li> </ul>		
<p><b>Testarea și refactorizarea programelor</b></p> <ul style="list-style-type: none"> <li>• Introducere în testarea programelor, niveluri de testare al programelor.</li> <li>• Testarea manuală și automată folosind PyUnit.</li> <li>• Introducere în paradigma dezvoltării ghidate de testare.</li> <li>• Inspectia și refactorizarea programelor.</li> </ul>		
<p><b>Pregătirea examenului</b></p> <ul style="list-style-type: none"> <li>• Recapitularea celor mai importante teme acoperite în cadrul cursului.</li> <li>• Prezentarea ghidului de examen.</li> </ul>		
<p><b>Bibliografie</b></p> <ol style="list-style-type: none"> <li>1. Kent Beck - <i>Test Driven Development: By Example</i>. Addison-Wesley Longman, 2002.</li> <li>2. Kleinberg and Tardos – <i>Algorithm Design</i>. Pearson Educational, 2014 (<a href="http://www.cs.princeton.edu/~wayne/kleinberg-tardos/">http://www.cs.princeton.edu/~wayne/kleinberg-tardos/</a>)</li> <li>3. Martin Fowler - <i>Refactoring. Improving the Design of Existing Code</i>. Addison-Wesley, 1999 (<a href="http://refactoring.com/catalog/index.html">http://refactoring.com/catalog/index.html</a>)</li> <li>4. <i>The Python language reference</i>. (<a href="https://docs.python.org/3/reference/index.html">https://docs.python.org/3/reference/index.html</a>)</li> </ol>		

5. <i>The Python standard library</i> . ( <a href="https://docs.python.org/3/library/index.html">https://docs.python.org/3/library/index.html</a> )		
6. <i>The Python tutorial</i> . ( <a href="https://docs.python.org/3/tutorial/index.html">https://docs.python.org/3/tutorial/index.html</a> )		
<b>8.2 Seminar / laborator</b>	<b>Metode de predare - învățare</b>	<b>Observații</b>
1. Prezentarea cursului și a limbajului Python	<ul style="list-style-type: none"> <li>• Expunere interactivă</li> <li>• Explicație</li> <li>• Conversație</li> <li>• Exemple</li> <li>• Demonstrație didactică</li> </ul>	
2. Recursivitatea. Complexitatea algoritmilor		
3. Căutare. Sortare		
4. Metode de rezolvare a problemelor: greedy, divide et impera, backtracking, programarea dinamică		
5. Programarea procedurală		
6. Programarea modulară		
7. Clase și obiecte		
8. Programarea bazată pe obiecte		
9. Proiectarea programelor. Arhitectura stratificată		
10. Pregătirea examenului		
<b>Bibliografie</b> 1. Kent Beck - <i>Test Driven Development: By Example</i> . Addison-Wesley Longman, 2002. 2. Kleinberg and Tardos - <i>Algorithm Design</i> . Pearson Educational, 2014 ( <a href="http://www.cs.princeton.edu/~wayne/kleinberg-tardos/">http://www.cs.princeton.edu/~wayne/kleinberg-tardos/</a> ) 3. Martin Fowler - <i>Refactoring. Improving the Design of Existing Code</i> . Addison-Wesley, 1999 ( <a href="http://refactoring.com/catalog/index.html">http://refactoring.com/catalog/index.html</a> ) 4. <i>The Python language reference</i> . ( <a href="https://docs.python.org/3/reference/index.html">https://docs.python.org/3/reference/index.html</a> ) 5. <i>The Python standard library</i> . ( <a href="https://docs.python.org/3/library/index.html">https://docs.python.org/3/library/index.html</a> ) 6. <i>The Python tutorial</i> . ( <a href="https://docs.python.org/3/tutorial/index.html">https://docs.python.org/3/tutorial/index.html</a> )		

## 9. Evaluare



















Tip activitate	9.1 Criterii de evaluare <sup>4</sup>	9.2 Metode de evaluare <sup>5</sup>	9.3 Pondere din nota finală
9.4 Curs	Corectitudinea și volumul cunoștințelor acumulate, demonstrate prin implementarea corectă a programelor în limbajul Python.	Examen scris	<b>30%</b>
9.5 Seminar/laborator	Competențe legate de proiectarea, implementarea, testarea și depanarea programelor Python.	Evaluare practică	<b>30%</b>
	Corectitudinea rezolvării temelor de laborator și a documentației aferente.	Portofoliul de programe și documentație	<b>40%</b>
9.6 Standard minim de promovare			
<ul style="list-style-type: none"> <li>• Studenții vor respecta standardele de integritate academică.</li> <li>• Studenții trebuie să demonstreze o bună înțelegere a conceptelor prezentate și exersate în timpul cursului, că pot utiliza aceste cunoștințe într-un mod coerent, că pot forma anumite conexiuni și pot utiliza cunoștințele și competențele dobândite pentru rezolvarea mai multor tipuri de probleme de programare.</li> <li>• Pentru a intra în procesul de evaluare desfășurat pe durata sesiunii de examene sau de restanțe, studenții trebuie să cumuleze un număr de 10 prezențe la seminar (din 14 posibile) și 12 prezențe la laborator (din 14 posibile).</li> <li>• Pentru a intra în procesul de evaluare desfășurat pe durata sesiunii de examene, studenții trebuie să aibă nota minimă de 5 la laboratorul desfășurat pe durata semestrului.</li> </ul>			

<sup>4</sup> Criteriile de evaluare trebuie să reflecte direct rezultatele învățării vizate la nivel de program de studii, respectiv la nivel de disciplină. Mai concret, se evaluează achizițiile de învățare menționate în rezultatele anticipate ale învățării.

<sup>5</sup> Se recomandă stabilirea atât a metodelor de evaluare finală, cât și a strategiei de evaluare pe parcurs.

- Pentru a promova examenul, studenții trebuie să obțină o notă mai mare sau egală cu 5 în cadrul activităților de laborator, a testului practic și a examenului scris.

## 10. Etichete ODD (Obiective de Dezvoltare Durabilă / Sustainable Development Goals)<sup>6</sup>

	<input type="radio"/>	Eticheta generală pentru Dezvoltare durabilă						
								
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<b>X</b>
								Nu se aplică nici o etichetă
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Data completării:

...

Semnătura titularului de curs

Assoc. Prof. PhD. Molnar Arthur

Semnătura titularului de seminar

Assoc. Prof. PhD. Molnar Arthur

Data avizării în departament:

...

Semnătura directorului de departament

Assoc. Prof. PhD. Sterca Adrian

<sup>6</sup> Selectați o singură etichetă, cea care, în conformitate cu [Procedura de aplicare a etichetelor ODD în procesul academic](#), se potrivește cel mai bine disciplinei. Dacă disciplina tratează tema dezvoltării durabile la modul general (de ex. prin prezentarea/introducerea cadrului general al dezvoltării durabile etc.) atunci se poate alocă eticheta generală de Dezvoltare Durabilă. Dacă niciuna dintre etichete nu descrie disciplina, selectați ultima opțiune: „Nu se aplică nici o etichetă”.