



Ministerul Educației, Culturii și Cercetării al Republicii Moldova
IP Centrul de Excelență în Informatică și Tehnologii Informaționale

"Aprob"



Directorul Centrului de Excelență în Informatică și
Tehnologii Informaționale

Vitalie Zavadschi

Curriculum modular

P.01.O.011 Programarea calculatorului

Specialitatea: **61230 Rețele de calculatoare**
Calificarea: **Tehnician pentru rețele de calculatoare**

Chișinău, 2022



**Curriculumul a fost elaborat în cadrul
I.P. Centrul de Excelență în Informatică
și Tehnologii Informaționale**

Autor:

Galina LUNCAȘU, grad didactic superior, Centrul de Excelență în Informatică și Tehnologii Informaționale.

Adoptat în baza Planului de învățământ, aprobat prin OMEC Nr.950/22, din 29.09.2022:

Gribineț Luminița, grad didactic unu, Centrul de Excelență în Informatică și Tehnologii Informaționale;

Șarapanovscaia Irina, grad didactic unu, Centrul de Excelență în Informatică și Tehnologii Informaționale;

Examinat la ședința catedrei Informatică II

Proces verbal Nr. 01 din 02 septembrie 2022

Șef catedră _____ L. Gribineț

Aprobat de:

Consiliul metodic-științific al Centrului de Excelență în Informatică și Tehnologii Informaționale

Proces verbal Nr. 01 din 14 septembrie 2022

Director adjunct _____ Obadă Liuba

Recenzenți:

1. S.A. MoldTelecom, or. Chișinău, str. bd. Ștefan cel Mare, 10
2. Starnet Soluții S.R.L, or. Chișinău, str. Columna, 170

Cuprins

I. Preliminarii	4
II. Motivația, utilitatea unității de curs pentru dezvoltarea profesională.....	4
III. Competențele profesionale specifice unității de curs	5
IV. Administrarea unității de curs	5
V. Unitățile de învățare	6
VI. Repartizarea orientativă a orelor pe unități de învățare	15
VII. Studiu individual ghidat de profesor	15
VIII. Lucrările practice recomandate	15
IX. Sugestii metodologice	16
X. Sugestii de evaluare a competențelor profesionale	18
XI. Resursele necesare pentru desfășurarea procesului de studii.....	20
XII. Resursele didactice recomandate elevilor	21

I. Preliminarii

Programarea este un proces creativ de elaborare a produselor-program necesare activităților realizate cu ajutorul calculatorului. Programarea presupune sarcini precum analiza problemei, împărțirea ei în subprobleme, gândirea și proiectarea de algoritmi necesari rezolvării problemei și implementarea algoritmilor într-un limbaj de programare ales. Procesul de programare astfel necesită adesea cunoștințe în mai multe subiecte diferite.

Statutul Curriculumului. Curriculumul modular “Programarea calculatorului” este un document normativ și obligatoriu pentru realizarea procesului de pregătire a tehnicienilor în învățământul profesional tehnic postsecundar, care vor elabora diverse aplicații în conformitate cu sarcinile de lucru.

Funcțiile Curriculumului. Funcțiile de bază ale Curriculumul sunt:

- act normativ al procesului de predare, învățare, evaluare și certificare în contextul unei pedagogii axate pe competențe;
- reper pentru proiectarea didactică și desfășurarea procesului educațional din perspectiva unei pedagogii axate pe competențe;
- componentă de bază pentru elaborarea strategiei de evaluare și certificare;
- orientare a procesului educațional spre formare de competențe la elevi;
- componentă fundamentală pentru elaborarea manualelor tipărite, manualelor electronice, ghidurilor metodologice, testelor de evaluare.

Beneficiarii Curriculumului. Curriculumul este destinat:

- profesorilor din instituțiile de învățământ profesional tehnic postsecundar;
- autorilor de manuale și ghiduri metodologice;
- elevilor care își fac studiile la specialitățile din domeniul de educație Tehnologia informației și a comunicațiilor;
- membrilor comisiilor pentru examenele de calificare;
- membrilor comisiilor de identificare, evaluare și recunoaștere a rezultatelor învățării, dobândite în contexte non-formale și informale.

Scopul modului este de transmite elevilor noțiunile fundamentale privind programarea și de a le forma deprinderi practice de lucru și abilitatea analitică de a rezolva probleme practice în cadrul unui mediu de programare de nivel înalt.

Pentru studierea modului nu este obligatoriu însușirea anumitor module.

II. Motivația, utilitatea unității de curs pentru dezvoltarea profesională

Studierea acestui modul va contribui la formarea și dezvoltarea de competențe profesionale ce corespund nivelului patru de calificare:

- cunoștințe factice, principii, procese și concepte generale din domeniul elaborării produselor program;
- abilități cognitive și practice necesare pentru elaborarea aplicațiilor de consolă conform tematicilor incluse;
- asumarea responsabilității pentru mentenanța de aplicații.

Competențele formate și dezvoltate în cadrul acestui modul vor fi necesare pentru studierea următoarelor module:

P.02.O.013 Asistență pentru programarea orientată pe obiecte

P.03.O.018 Tehnologii avansate de programare

P.04.O.021 Testarea și depănarea produselor program

P.04.O.022 Dezvoltarea aplicațiilor pentru dispozitive mobile

De asemenea, ele vor fi de un real folos în activitatea profesională a asistentului programator, în special, în ocupațiile legate de gestiunea produselor-program utilizate în companii.

III. Competențele profesionale specifice unității de curs

În cadrul modului vor fi formate și dezvoltate următoarele competențe profesionale specifice:

- CS1. Translarea algoritmilor de decizie/ciclici în cadrul unei aplicații de consolă;
- CS2. Elaborarea algoritmilor de rezolvare a problemelor pentru diverse tipuri de date;
- CS3. Organizarea programelor la nivel de subprograme;
- CS4. Prelucrarea structurilor de date în cadrul aplicațiilor de consolă;
- CS5. Gestionarea eficientă a memoriei interne a calculatorului;
- CS6. Utilizarea tehnicilor de programare pentru problemele întâlnite în activitatea profesională.

IV. Administrarea unității de curs

Semestrul	Numărul de ore contact direct pe semestrul de studiu				Forma de evaluare	Numărul de credite
	Total	Contact direct		Ore de studiu individual		
		IT	IP			
I	210	60	90	60	Examen	7

V. Unitățile de învățare

Unități de competență	Unități de conținut	Abilități
1. Elementele de bază ale limbajului de programare		
UC1. Editarea fragmentelor de program.	Vocabularul limbajului. Unități lexicale. Conceptul de dată. Definirea tipurilor de date. Operatori. Structura generală a unui program. Variabile, constante. - declarare; - inițializare. Operatori. Precedența operatorilor. Conversii.	A1. Scrierea simbolurilor speciale în limbajul de programare. A2. Identificarea cuvintelor cheie ale limbajului de programare. A3. Scrierea identificatorilor. A4. Declararea variabilelor/constantelor conform tipului de dată solicitat. A5. Inițializarea variabilelor/constantelor cu valori prestabilite. A6. Selectarea operatorilor în dependență de specificul tipului de dată utilizat. A7. Scrierea expresiilor aritmetice în limbajul de programare. A8. Identificarea operatorilor și operanzilor. A9. Utilizarea operatorilor și operanzilor la prelucrarea datelor. A10. Utilizarea operatorilor conform specificațiilor propuse. A11. Determinare a compatibilității componentelor instrucțiunilor din punct de vedere al atribuirii. A12. Realizarea conversiilor de date conform specificațiilor propuse. A13. Identificarea părților componente ale unui program.
UC2. Elaborare a de programe elementare.	Operații de ieșire la nivel de consolă. Operații de intrare la nivel de consolă.	A14. Afișarea valorilor de tip constante. A15. Afișarea valorilor de tip variabilă. A16. Afișarea datelor cu format. A17. Afișarea datelor conform specificațiilor propuse. A18. Citirea datelor conform specificațiilor propuse.
UC3. Scrierea programelor în cadrul mediului de programare.	Interfața mediului de programare. Tipuri de instrucțiuni: - vide; - simple;	A19. Lansarea mediului de programare. A20. Salvarea unei aplicații de consolă. A21. Deschiderea unei aplicații de consolă. A22. Compilarea unei aplicații de consolă.

Unități de competență	Unități de conținut	Abilități
	- compuse.	A23. Executarea unei aplicații de consolă. A24. Personalizarea mediului de programare. A25. Interpretarea rezultatelor unei aplicații de consolă. A26. Testarea unei aplicații de consolă. A27. Evitarea erorilor în cadrul unei aplicații de consolă. A28. Evaluarea expresiilor limbajului de programare. A29. Evaluarea expresiilor propuse și determinarea tipului lor.
UC4. Utilizarea instrucțiunilor de decizie în cadrul aplicațiilor de consolă.	Instrucțiuni de decizie. Instrucțiuni de decizie multiplă.	A30. Identificarea părților componente ale unei instrucțiuni de decizie. A31. Translarea algoritmilor cu decizie unică în limbajul de programare. A32. Translarea algoritmilor cu decizie dublă în limbajul de programare. A33. Translarea algoritmilor cu decizie triplă în limbajul de programare. A34. Translarea algoritmilor cu decizie multiplă în limbajul de programare. A35. Elaborarea algoritmilor de decizie conform specificațiilor propuse. A36. Implementarea algoritmilor de decizie în limbajul de programare.
UC5. Utilizarea ciclurilor în cadrul aplicațiilor de consolă.	Instrucțiunea ciclică. Particularități ale instrucțiunilor ciclice: - întrerupere forțată; - reevaluarea condiției de ciclare.	A37. Identificarea părților componente ale unei instrucțiuni ciclice. A38. Elaborarea algoritmilor ciclici conform specificațiilor propuse. A39. Elaborarea algoritmilor ciclici cu un număr cunoscut de pași. A40. Elaborarea algoritmilor ciclici cu un număr necunoscut de pași. A41. Elaborarea algoritmilor ciclici cu întrerupere forțată. A42. Elaborarea algoritmilor ciclici cu reevaluarea condiției de ciclare. A43. Translarea algoritmilor ciclici în limbajul de programare. A44. Implementarea algoritmilor ciclici în limbajul de programare.
UC6. Prelucrarea obiectelor în cadrul aplicațiilor de consolă.	Obiecte și referințe. Supraîncărcarea operatorilor pentru obiecte	A45. Declararea obiectelor A46. Colectarea de gunoae A47. Transmiterea de parametri A48. Supraîncărcarea operatorilor

Unități de competență	Unități de conținut	Abilități
UC7. Prelucrare tablourilor în cadrul aplicațiilor de consolă.	Tablouri unidimensionale. Tablouri bidimensionale.	A49. Declararea tablourilor unidimensionale/ bidimensionale. A50. Inițializarea elementelor unui tablou unidimensional/bidimensional. A51. Citirea elementelor tablourilor unidimensionale/ bidimensionale. A52. Afișarea elementelor tablourilor unidimensionale/ bidimensionale. A53. Prelucrarea elementară a tablourilor unidimensionale/ bidimensionale. A54. Permutarea elementelor tabloului unidimensional/ bidimensionale. A55. Eliminarea elementelor din tablou. A56. Interclasarea tablourilor unidimensionale.
UC8. Prelucrare șirurilor de caractere în cadrul aplicațiilor de consolă.	Șiruri de caractere. Funcții standard pentru prelucrarea șirurilor de caractere. Funcții pentru conversii de date.	A57. Declararea șirurilor de caractere. A58. Utilizarea șirurilor de caractere în cadrul aplicațiilor. A59. Concatenarea șirurilor de caractere. A60. Compararea șirurilor de caractere. A61. Căutarea unui șir de caractere. A62. Ștergerea subșirurilor de caractere. A63. Inserarea subșirurilor de caractere. A64. Utilizarea funcțiilor standard la prelucrarea șirurilor de caractere. A65. Conversia datelor din/în șir de caractere.
UC9. Utilizarea subprogramelor în cadrul aplicațiilor de consolă	Noțiune de subprograme. Domenii de vizibilitate: – structura de bloc a programelor; – variabile globale și variabile locale. Efecte colaterale: – cauzele efectelor colaterale; – modul de evitare a efectelor colaterale. Supraîncărcarea metodelor	A66. Utilizarea terminologiei specifice subprogramelor. A67. Declararea subprogramelor. A68. Declararea anticipată a subprogramelor. A69. Apelarea subprogramelor . A70. Elaborarea subprogramelor. A71. Prelucrarea datelor utilizate în procesul de studiu la nivel de subprograme. A72. Comunicarea între subprograme prin intermediul variabilelor globale/parametrilor. A73. Evitarea efectelor colaterale. A74. Utilizarea subprogramelor definite de către utilizator în cadrul programelor. A75. Supraîncărcarea numelor subprogramelor

Unități de competență	Unități de conținut	Abilități
		A76. Elaborarea algoritmilor la nivel de subprograme. A77. Implementarea algoritmilor bazați pe subprograme în limbajul de programare.
2. Analiza eficienței algoritmilor și Structuri de date		
UC10. Estimarea complexității algoritmilor	Algoritmi. Analiza algoritmilor: - necesarul de memorie; - complexitatea temporală a algoritmului; - clasificarea algoritmilor.	A79. Utilizarea corectă terminologiei aferente complexității algoritmilor. A80. Estimarea complexității algoritmilor utilizați în procesul de studiu. A81. Estimarea necesarului de memorie necesar pentru execuția unui program.
UC11. Sortarea datelor structurate în cadrul aplicațiilor de consolă	Metode de sortare: - Sortarea prin inserție (Insertion sort); - Sortarea prin selecție (Select sort); - Sortarea prin metoda bulelor (Bubble sort); - Sortarea prin interclasare (MergeSort); - Sortarea rapidă (QuickSort);	A78. Selectarea algoritmilor de sortare a datelor structurate. A79. Prezentarea situațiilor de aplicare a metodelor de sortare. A80. Detalierea algoritmilor de sortare a datelor structurate. A81. Translarea algoritmilor de sortare în limbajul de programare. A82. Implementarea algoritmilor de sortare în limbajul de programare.
UC12. Utilizarea subprogramelor recursive în cadrul aplicațiilor de consolă	Recursivitate	A83. Descrierea structurii unui subprogram recursiv/iterativ. A84. Argumentarea necesității utilizării unui algoritm recursiv/iterativ. A85. Elaborarea algoritmilor recursivi/iterativi pentru problemele din activitatea profesională. A86. Translarea algoritmilor recursivi/iterativi în limbajul de programare. A87. Implementarea algoritmilor recursivi/iterativi în limbajul de programare.
UC11. Prelucrarea tipurilor dinamice de date de tip listă în cadrul aplicațiilor de consolă	Structuri de date de tip Liste. Operații specifice.	A88. Declararea unei liste. A89. Crearea unei liste, conform specificațiilor propuse. A90. Afișarea datelor unei liste, conform specificațiilor propuse. A91. Inserarea datelor noi într-o listă. A92. Ștergerea unui element din listă, conform specificațiilor propuse.

Unități de competență	Unități de conținut	Abilități
		A93. Căutarea datelor unei liste, conform specificațiilor propuse. A94. Elaborarea algoritmilor pentru structura de date de tip listă. A95. Translarea algoritmilor pentru tipuri de date de tip listă în limbajul de programare. A96. Implementarea algoritmilor pentru tipuri de date de tip listă în limbajul de programare.
UC12. Prelucrarea tipurilor dinamice de date de tip stivă/coadă în cadrul aplicațiilor de consolă	Structuri de date prestabilite: - Stivă; - Coadă. Operații specifice .	A82. Crearea unei structuri de date de tip stivă, coadă. A83. Afișarea datelor unei structuri de date de tip stivă, coadă. A84. Căutarea datelor unei structuri de date de tip stivă, coadă. A85. Eliminarea datelor în cadrul unei structuri de date de tip stivă, coadă. A86. Adăugarea datelor în cadrul unei structuri de date de tip stivă, coadă. A87. Distrugerea unei structuri de date de tip stivă, coadă. A88. Elaborarea algoritmilor pentru tipuri de date de tip stivă, coadă. A89. Translarea algoritmilor pentru tipuri de date de tip stivă, coadă în limbajul de programare. A90. Implementarea algoritmilor pentru tipuri de date de tip stivă, coadă în limbajul de programare.
UC13. Prelucrarea tipurilor dinamice de date de tip set/map în cadrul aplicațiilor de consolă	Structuri de date prestabilite: - Set-uri; - Map-uri. Operații specifice .	A91. Crearea unei structuri de date de tip set, map. A92. Afișarea datelor unei structuri de date de tip set, map. A93. Căutarea datelor unei structuri de date de tip set, map. A94. Eliminarea datelor în cadrul unei structuri de date de tip set, map. A95. Adăugarea datelor în cadrul unei structuri de date de tip set, map. A96. Distrugerea unei structuri de date de tip set, map. A97. Elaborarea algoritmilor pentru tipuri de date de tip set, map. A98. Translarea algoritmilor pentru tipuri de date de tip set, map în limbajul de programare.

Unități de competență	Unități de conținut	Abilități
		A99. Implementarea algoritmilor pentru tipuri de date de tip set, map în limbajul de programare.
UC14. Prelucrarea tipurilor de date de tip arbore binar în cadrul aplicațiilor de consolă	Arbori binari. Arbori binari de căutare. Operații specifice.	A100. Crearea unei structuri de tip arbore binar. A101. Afișarea datelor unei structuri de date de tip arbore binar. A102. Căutarea datelor unei structuri de date de tip arbore binar. A103. Eliminarea datelor în cadrul unei structuri de date de tip arbore binar. A104. Adăugarea datelor în cadrul unei structuri de date de tip arbore binar. A105. Distrugerea unei structuri de date de tip arbore binar. A106. Parcurgerea arborilor binari prin intermediul metodelor: în lățime, în lungime, în adâncime. A107. Elaborarea algoritmilor pentru tipuri de date de tip arbore binar. A108. Translarea algoritmilor pentru tipuri de date de tip arbore binar în limbajul de programare. A109. Implementarea algoritmilor pentru tipuri de date de tip arbore binar în limbajul de programare.
UC15. Prelucrarea tipurilor de date de tip tabelului de reprezentarea în cadrul aplicațiilor de consolă	Tabele de repartizare (hashtable)	A110. Adăugarea datelor în cadrul unei structuri de date de tip tabel de repartizare. A111. Eliminarea datelor în cadrul unei structuri de date de tip tabel de repartizare. A112. Căutarea datelor în cadrul unei structuri de date de tip tabel de repartizare. A113. Elaborarea algoritmilor pentru tipuri de date de tip tabel de repartizare. A114. Translarea algoritmilor pentru tipuri de date de tip tabel de repartizare în limbajul de programare. A115. Implementarea algoritmilor pentru tipuri de date de tip tabel de repartizare în limbajul de programare.

Unități de competență	Unități de conținut	Abilități
UC16. Utilizarea algoritmilor bazați pe metode de reprezentare a grafurilor în cadrul aplicațiilor de consolă	Grafuri. Terminologie și proprietăți. Reprezentarea grafurilor în memoria calculatorului.	A116. Utilizarea terminologiei specifice a teoriei grafurilor. A117. Prezentarea problemelor din viața reală, care pot fi soluționate cu ajutorul teoriei grafurilor. A118. Reprezentarea unui graf în memoria calculatorului prin matricea de incidență, matricea de adiacență, listei. A119. Descrierea algoritmilor de introducere, extragere și transformare a diferitelor forme de reprezentare internă a grafurilor A120. Elaborarea algoritmilor bazați pe proprietățile grafurilor. A121. Translarea algoritmilor de introducere, extragere și transformare a diferitelor forme de reprezentare internă a grafurilor în limbajul de programare. A122. Implementarea algoritmilor de introducere, extragere și transformare a diferitelor forme de reprezentare internă a grafurilor în limbajul de programare.
UC17. Utilizarea algoritmilor de calcul a drumului minim într-un graf	Algoritmi de determinare a drumului minim Extragerea arborelui minimal într-un graf:	A123. Prezentarea situațiilor de aplicare drumului minim. A124. Aplicarea algoritmilor de determinare a drumului minim. A125. Prezentarea situațiilor de aplicare a arborelui minimal. A126. Aplicarea algoritmilor de extragere a arborelui minimal. A127. Translarea algoritmilor de determinare a drumului minim în limbajul de programare. A128. Translarea algoritmilor de extragere a arborelui minimal în limbajul de programare. A129. Implementarea algoritmilor de determinare a drumului minim în limbajul de programare. A130. Implementarea algoritmilor de extragere a arborelui minimal în limbajul de programare.

Unități de competență	Unități de conținut	Abilități
3. Metode de elaborare a algoritmilor		
UC18. Utilizarea algoritmilor bazați pe metoda reluării în cadrul aplicațiilor de consolă	Metoda reluării (Backtracking): – tehnica de reluare; – domeniile de aplicare a metodei reluării.	A131. Descrierea algoritmului bazat pe metoda reluării. A132. Prezentarea situațiilor de aplicare a metodei reluării. A133. Elaborarea algoritmilor bazați pe metoda reluării conform specificațiilor propuse. A134. Translarea algoritmilor bazați pe metoda reluării în limbajul de programare. A135. Implementarea algoritmilor bazați pe metoda reluării în limbajul de programare.
UC19. Utilizarea algoritmilor bazați pe metoda desparte și stăpânește în cadrul aplicațiilor de consolă	Metoda desparte și stăpânește (Divide et impera): – tehnica desparte și stăpânește; – domeniile de aplicare a metodei desparte și stăpânește.	A136. Descrierea algoritmului bazat pe metoda desparte și stăpânește. A137. Prezentarea situațiilor de aplicare a metodei desparte și stăpânește. A138. Elaborarea algoritmilor bazați pe metoda desparte și stăpânește conform specificațiilor propuse. A139. Translarea algoritmilor bazați pe metoda desparte și stăpânește în limbajul de programare. A140. Implementarea algoritmilor bazați pe metoda desparte și stăpânește în limbajul de programare.
UC20. Utilizarea algoritmilor bazați pe metoda Greedy în cadrul aplicațiilor de consolă	Metoda Greedy: – tehnica Greedy; – domeniile de aplicare a metodei Greedy.	A141. Descrierea algoritmului bazat pe metoda Greedy. A142. Prezentarea situațiilor de aplicare a metodei Greedy. A143. Elaborarea algoritmilor bazați pe metoda Greedy conform specificațiilor propuse. A144. Translarea algoritmilor bazați pe metoda Greedy în limbajul de programare. A145. Implementarea algoritmilor bazați pe metoda Greedy în limbajul de programare.
UC21. Utilizarea	Metoda programării dinamice:	A146. Descrierea algoritmului bazat pe metoda programarea dinamică.

Unități de competență	Unități de conținut	Abilități
algoritmilor bazați pe metoda programării dinamice în cadrul aplicațiilor de consolă	- tehnica programarea dinamică; - domeniile de aplicare a metodei programării dinamice.	A147. Prezentarea situațiilor de aplicare a metodei programarea dinamică. A148. Elaborarea algoritmilor bazați pe metoda programarea dinamică conform specificațiilor propuse. A149. Translarea algoritmilor bazați pe metoda programarea dinamică în limbajul de programare. A150. Implementarea algoritmilor bazați pe metoda programarea dinamică în limbajul de programare.

VI. Repartizarea orientativă a orelor pe unități de învățare

Nr. crt.	Unități de învățare	Ore de contact direct			Ore de studiu individual	Ore de contact direct după locul de instruire			
		Total	Contact direct			I/I		I/A	
			IT	IP		IT	IP	IT	IP
1.	Elementele de bază ale limbajului de programare	86	32	24	30	32	24	-	-
2.	Analiza eficienței algoritmilor și Structuri de date	76	20	26	30	20	26	-	-
3.	Metode de elaborare a algoritmilor	48	8	10	30	8	10	-	-
	Total	210	60	60	90	60	60	-	-

VII. Studiu individual ghidat de profesor

Materii pentru studiul individual	Produce de elaborat	Modalități de evaluare	Termeni de realizare
1. Elementele de bază ale limbajului de programare			
Tablouri. Șiruri de caractere.	Portofoliu: Set de aplicații de consolă pentru prelucrarea tablourilor și a șirurilor de caractere.	Prezentarea portofoliului. Demonstrarea pe calculator.	Săptămâna 7
2. Analiza eficienței algoritmilor și Structuri de date			
Structuri de date	Portofoliu: Set de aplicații de consolă pentru prelucrarea structurilor de date.	Prezentarea portofoliului. Demonstrarea pe calculator.	Săptămâna 12
3. Metode de elaborare a algoritmilor			
Tehnici de programare	Portofoliu: Set de aplicații de consolă bazate pe implementare a tehnicilor de programare.	Prezentarea portofoliului. Demonstrarea pe calculator.	Săptămâna 14

VIII. Lucrările practice recomandate

Lucrările practice vor fi efectuate în formă de lucrări de laborator. Tematica lucrărilor de laborator va include:

1. Introducerea și extragerea datelor la nivel de consolă.
2. Evaluarea expresiilor aritmetice.

3. Implementarea algoritmilor de decizie.
4. Implementarea algoritmilor ciclici.
5. Referințe
6. Prelucrarea tablourilor
7. Prelucrarea șirurilor de caractere.
8. Subprograme
9. Recursivitate
10. Sortarea datelor.
11. Prelucrarea datelor în cadrul structurilor de date
12. Introducerea și extragerea grafurilor, transformarea a formelor de reprezentare internă a acestora.
13. Implementarea algoritmilor bazați pe metoda de elaborare a algoritmilor

IX. Sugestii metodologice

Elementul de bază al Curriculumului sunt competențele ce trebuie formate și dezvoltate în procesul de formare profesională. Acestea vor fi formate prin organizarea eficientă a procesului de instruire. Pentru aceasta sunt necesare două condiții:

1. Organizarea activităților. Pentru buna organizare a procesului didactic ambii participanți necesită de a-și organiza activitățile. De modul cum sunt organizate acestea depinde în mare măsură nivelul de formare a competențelor. În această ordine de idei, în procesul de organizare a activităților se vor asigura:

- condiții optime pentru buna colaborare dintre elev și profesor;
- un set de procese care duc la îmbunătățirea relațiilor dintre părți;
- un nivel de implicare a părților acționând în baza unor reguli și acțiuni prestabilite.

2. Selectarea adecvată a metodelor de instruire. Se recomandă utilizarea metodelor de instruire precum:

Simularea și modelarea. Simularea este utilizată pentru prezentarea la faza inițială a unor concepte, oferind posibilitatea de ghidare a activității elevului în bază de situații practice. Prin intermediul acestei metode se pot reda, prin analogie, diverse situații, raționamente, care pot să reprezinte relații dintre obiecte, fenomene, procese etc.

Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Structura generală a unui program;
- Conceptul de dată. Definirea tipurilor de date;
- Tipuri de instrucțiuni;
- Instrucțiuni ciclice,
- Metode de sortare;
- Referințe ;
- Subprograme recursive;
- Structuri de date;
- Operații specifice structurilor de date;
- Arbori binari de căutare.

Problematicizarea mai poate fi denumită și predare prin rezolvare de probleme sau predare productivă de probleme. Conform acestei metode instruitului este pus în fața unor dificultăți create în mod deliberat, și prin depășirea lor învață ceva nou. „Punctul forte” al metodei îl constituie situația-problemă. Din această cauză este necesar de a formula corect situația. La crearea situație de tip problemă se va ține cont de următoarele caracteristici:

- A. Situația trebuie să prezinte o dificultate pentru instruit, iar pentru a găsi soluția, acesta se va confrunta cu efort de gândire;
- B. Situația trebuie să prezinte interes, astfel încât acesta să acționeze spre a rezolva problema;
- C. Situația trebuie să orienteze activitatea instruitului spre a rezolva problema și de al cointeresa pe acesta de a dobândi noi cunoștințe;
- D. Rezolvarea situației nu va fi posibilă fără a apela la resurselor recent dobândite.

Prin intermediul situației create, instruitul este cointerestat de a studia, analiza și a participa la rezolvarea problemei. Aplicarea acestei metode presupune parcurgerea a patru etape:

1. Formularea problemei – este descrisă situația problemă, explicarea, după necesitate a diferitor puncte cheie, care ar permite instruitului să perceapă problema;
2. Studiarea problemei – se lucrează în mod independent, sunt reactualizate anumite resurse;
3. Determinarea soluției – în cadrul acestei etape sunt pregătite resursele necesare, se descoperă mijloacele care duc la rezolvarea problemei și este analizat modul de aplicare a acestora în determinarea soluției;
4. Obținerea rezultatului final – se analizează rezultatul obținut și formate anumite concluzii.

Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Instrucțiuni de decizie;
- Instrucțiuni ciclice;
- Tablouri
- Efecte colaterale;
- Supraîncărcarea funcțiilor;
- Analiza algoritmilor;
- Abordări recursive și iterative;
- Metode de programare.

Algoritmizarea reprezintă o metodă de predare-învățare bazată pe utilizarea și valorificarea algoritmilor în procesul de instruire. Algoritmii de instruire se reprezintă sub forma unui grup de scheme, unui set de operații, iar prin parcurgerea lor într-o ordine bine stabilită duc la rezolvarea unui set de probleme caracteristice unei familii de situații. În rezultatul aplicării acestei metode se va oferi posibilitatea elevului de a elabora treptat propriile scheme, aplicabile în diferite circumstanțe didactice. Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Instrucțiuni de decizie;
- Instrucțiuni ciclice;
- Tablouri;
- Metode de sortare;
- Subprograme recursive;
- Metode de programare.

Instruirea asistată de calculator este o metodă didactică care valorifică principiile de modelare și analiză cibernetică. Prin intermediul calculatorului se pune la dispoziția elevului un set de probleme, care necesită a fi analizate, completate sau elaborate. Utilizarea metodei va oferi posibilitatea de

- organizare a informației conform cerințelor programei adaptabile la capacitățile fiecărui elev;
- stimulare cognitivă a elevului prin secvențe didactice și întrebări ce vizează depistarea unor lacune, probleme, situații-problemă;

- rezolvare a sarcinilor didactice prezentate anterior prin reactivarea sau obținerea informațiilor necesare de la resursele informatice apelate prin intermediul calculatorului;
- realizare a unor sinteze recapitulative după parcurgerea unor teme, module de studiu, lecții;
- asigurare a unor exerciții suplimentare de stimulare a creativității elevului.

Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Interfața mediului de programare;
- Operații de intrare/ieșire la nivel de consolă;
- Metode de sortare;
- Subprograme recursive;
- Metode de programare;

precum și în cadrul orelor de laborator.

Metoda studiul de caz valorifică o situație reală care se analizează și se rezolvă. Așa cum problemele rezolvate în stilul orientat pe obiecte au un grad sporit de dificultate, sunt cazuri când este necesar de a prezenta elevului probleme deja rezolvate. Avantajul metodei, constă în faptul că fiecare dintre elev își va aduce aportul la analiza și rezolvarea problemei. În utilizarea acestei metode se conturează câteva etape: 1) Selectarea și prezentarea cazului; 2) Organizarea echipelor de lucru; 3) Prelucrarea și conceptualizarea; 4) Structurarea finală a studiului.

Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Metode de sortare;
- Subprograme;
- Structuri de date prestabilite;
- Grafuri. Terminologie și proprietăți;
- Reprezentarea grafurilor în memoria calculatorului;
- Metode de programare.

Instruirea prin proiecte reprezintă o modalitate de instruire/autoinstruire grație căreia elevii, dar mai ales elevii efectuează o cercetare orientată spre obiective practice și finalizată într-un produs ce poate fi un obiect, un aparat, o instalație, o culegere tematică, un album, o lucrare științifică etc. Această metodă se recomandă pentru predarea-învățarea-evaluarea următoarelor unități de conținut:

- Metode de sortare;
- Structuri de date;
- Subprograme;
- Grafuri. Terminologie și proprietăți;
- Reprezentarea grafurilor în memoria calculatorului;
- Metode de programare.

X. Sugestii de evaluare a competențelor profesionale

Evaluarea competențelor profesionale este procesul prin care sunt colectate și analizate dovezile necesare pentru judecarea competenței în raport cu cerințele calificării profesionale. Calificarea profesională este documentul în care se descriu rezultatele învățării în concordanță cu cerințele pieței muncii, specificate în standardul ocupațional/ profilul ocupațional. Evaluarea competențelor profesionale este un proces complet diferit de sistemul tradițional de evaluare a cunoștințelor. Evaluarea competențelor profesionale este un proces care presupune consultarea și colaborarea dintre elev și profesor. Evaluarea

competențelor are loc prin furnizarea de către elev a dovezilor de competență care sunt interpretate de către profesor. Dovezile de competență acumulate sunt rezultate considerate parțiale și atât elevul cât și profesorul pot solicita clarificări suplimentare.

Procedura de evaluare a competențelor profesionale pentru modulul *Programarea calculatorului*, va oferi elevilor posibilitatea de a-și demonstra atât cunoștințele teoretice și practice. Metodele folosite în procesul de evaluare vor evidenția cunoștințele și deprinderile necesare pentru efectuarea activităților de muncă și, mai ales, capacitatea elevului de a obține rezultatele practice așteptate.

Activitățile de evaluare vor fi orientate spre motivarea elevilor și obținerea unui feedback continuu, fapt ce va permite corectarea operativă a procesului de învățare, stimularea autoevaluării și a evaluării reciproce, evidențierea succeselor, implementarea evaluării selective sau individuale. Pentru a eficientiza procesele de evaluare, înainte de a demara evaluările, cadrul didactic va aduce la cunoștință elevilor tematica lucrărilor, modul de evaluare (bareme/grile/criterii de notare) și condițiile de realizare a fiecărei evaluări.

Evaluarea curentă/formativă se va realiza prin diverse modalități: observarea comportamentului elevului, analiza rezultatelor activității elevului, discuția/conversația, prezentarea proiectelor individuale de activitate. Prin evaluarea curentă/formativă, cadrele didactice informează elevul despre nivelul de performanță; îl motivează să se implice în dobândirea competențelor profesionale.

Evaluarea sumativă se realizează la finele modulului în baza simulării în atelier a unei situații de problemă din contexte profesionale variate, care solicită elevului demonstrarea competenței profesionale. Cadrele didactice vor elabora sarcini prin care vor orienta comportamentul profesional al elevului spre demonstrarea sistemului de cunoștințe și abilități. În acest scop, vor fi clar stabiliți indicatorii și descriptorii de performanță ai procesului și produsului realizat de către elev.

Portofoliul reprezintă o metodă complexă de evaluare în care un rezultat al evaluării este elaborat pe baza aplicării unui ansamblu variat de probe și instrumente de evaluare. Portofoliul, de regulă este realizat pe o perioadă mai îndelungată (în decursul mai multor ore). Conținutul unui portofoliu este reprezentat de rezultatele la: lucrări practice, studiul individual, investigații, referate și proiecte, observarea sistematică la clasă, autoevaluarea elevului, chestionare de atitudini etc. Alegerea elementelor ce formează portofoliul este realizată de către profesor (astfel încât acestea să ofere informații concludente privind pregătirea, evoluția, atitudinea elevului) sau chiar de către elev (pe considerente de performanță, preferințe etc.). Structurarea evaluării sub forma de portofoliu se dovedește deosebit de utilă, atât pentru profesor, cât și pentru elev sau părinții acestuia. Pentru a realiza o evaluare pe bază de portofoliu, profesorul:

- va comunica elevilor intenția de a realiza un portofoliu, adaptând instrumentele de evaluare ce constituie "centrul de greutate" ale portofoliului la specificul unității de învățare;
- va alege componentele ce formează portofoliul, dând și elevului posibilitatea de a adăuga piese pe care le consideră relevante pentru activitatea sa;
- va evalua separat fiecare piesă a portofoliului în momentul realizării ei, dar va asigura și un sistem de criterii pe baza cărora să realizeze evaluarea globală și finală a portofoliului;
- va pune în evidență evoluția elevului, particularitățile de exprimare și de raportare a acestuia la aria vizată;
- va integra rezultatul evaluării portofoliului în sistemul general de notare.

Competențele elevului se manifestă prin produse concrete, care sunt analizate de către profesor în raport cu aspectele critice stabilite pentru unitate/unitățile de competență pentru care este evaluat. Dovezile de competență sunt informațiile produse de un elev din care rezultă că îndeplinește toate aspectele descrise de unitatea/unitățile de competență pentru care este evaluat, respectiv are cunoștințele și deprinderile necesare.

Evaluarea nivelului de dezvoltare a competențelor în cadrul orelor:

- **teoretice** se va realiza prin teste, exemple de aplicare a cunoștințelor teoretice în practică, machete etc.;
- **practice / de laborator** se va realiza prin elaborarea de către elev, în termeni concreți, a aplicațiilor având la bază unitățile de conținut studiate în cadrul orelor teoretice precum și abilitățile anterior dezvoltate;
- **de studiu individual** se va realiza prin studierea de către elev a materialelor suplimentare decât cele oferite în cadrul orelor de tip contact direct și prezentarea de portofolii pentru anumite unități de conținut și aplicații complexe prin care elevul își va demonstra abilitățile formate.

Probe de evaluare a competențelor, în baza situațiilor de problemă de la viitoarele locuri de muncă:

- elaborarea aplicațiilor de consolă conform specificațiilor propuse;
- evaluarea aplicațiilor de consolă elaborate;
- modificarea aplicațiilor de consolă conform specificațiilor propuse;
- testarea aplicațiilor de consolă elaborate.

În calitate de **produse pentru măsurarea competențelor** se vor folosi:

- aplicații de consolă elaborate conform specificațiilor propuse;
- algoritmi elaborați conform specificațiilor propuse.

Criteriile de evaluare a produselor pentru măsurarea competenței vor include:

- Utilizarea corectă a instrucțiunilor limbajului de programare.
- Corectitudinea algoritmilor elaborați.
- Fundamentarea deciziilor.
- Ținuta lingvistică.
- Respectarea termenilor de elaborare.

Forma finală de evaluare a unității de curs este examenul. Pentru a fi admis la examen elevul trebuie să susțină evaluările curente, să însușească materialul teoretic și să îndeplinească sarcinile propuse pentru activitățile practice, precum și studiul individual.

Nota finală va fi formată: 60% - media notelor curente și 40% - nota de la examen.

XI. Resursele necesare pentru desfășurarea procesului de studii

Cerințe față de sălile de curs	
Pentru orele teoretice	Cabinet de informatică cu 15 calculatoare Proiector
Pentru orele practice / de laborator	Laborator de informatică care asigură fiecărui elev un calculator
Cerințe tehnice	
Parametri tehnici minimi ale calculatorului	Procesor: 2 GHz Memorie operativă: 4 GB Unitate de stocare: 500 GB Afișaj și grafică: size: 22", resolution: 1366x768 Network: Ethernet, 100 Mb
Software	Sistem de Operare: Microsoft Windows , Mac, Linux Medii de dezvoltare: IntelliJ IDEA/Eclipse/ NetBeans/ Visual Studio

XII. Resursele didactice recomandate elevilor

Nr. crt.	Denumirea resursei
1.	E. Cerchez, M. Șerban, Programarea în limbajul C/C++ pentru liceu, 2005. http://cnvranceanu.ro/blog/wp-content/uploads/2014/11/cerchez-verde-m.pdf
2.	G. Vasilache, S. Gîncu Culegere de probleme la informatica, Chișinău, 2012. http://en.calameo.com/read/002801569a611d413be1c
3.	Daniel Danciu, George Mardale. ARTA PROGRAMĂRII ÎN JAVA, Vol1, Vol 2, Albastra, 2003
4.	T.Sorin, V.D.Huțanu Bazele programării in Java, L&S Info-mat, 2018
5.	I.Atanasiu , B.Costinescu, O.A.Dragoi , F.I.Popovici Limbajul Java. O perspectiva pragmatică. Editura Agora, Tg. Mureș, 2004.
6.	Introducere în Programarea .Net Framework. https://dl-manual.com/doc/manual-c-6dow7pr92z58
7.	Algoritmi și structuri de date. Note de Curs. https://ru.scribd.com/document/103258372/Algoritmi-Si-Structuri-de-Date
8.	S. Cataranciuc, TEORIA GRAFURILOR IN PROBLEME SI APLICATII, Chișinău, 2004. http://www.math.md/studlib/matematica/teoria_graf.html
9.	A. Ruceanu, Curs https://www.ruceanu.ro/adrian/
10.	https://www.codejava.net/
11.	http://www.oracle.com/technetwork/java/index.html
12.	Cărți pentru programatori. http://www.infoarena.ro/blog/carti