Ministerul Educaţiei şi Cercetării al Republicii Moldova
Centrul de Excelență în Informatică şi Tehnologii Informaționale

"Aprob"
Directorul Centrului de Excelență în
Informatică şi Tehnologii Informaționale

_____ Vitalie Zavadschi

Curriculumul modular
F.04.O.015 Asistență pentru programarea orientată pe obiecte

Specialitatea: **61230 Reţele de calculatoare**
Calificarea: **Tehnician pentru reţele de calculatoare**

Chişinău 2022

Aprobat de:

Consiliul metodico-ştiinţific al Centrului de Excelenţă în Informatică şi Tehnologii Informaţionale, proces verbal Nr. 3 din 31.10.2022

Director adjunct _____
Obadă Liuba

Autori:

Gîncu Silviu, doctor în pedagogie, grad didactic superior.

Pîrvan Evgheni, grad didactic superior, Colegiul „Iulia Hașdeu” din Cahul.

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

Actualizat în baza Planului de învățământ nr. SC-14/20, aprobat prin OMEC Nr.1310, din 23.11.2020:

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

Recenzenți:

1. *Bejenari Constanța*, manager direcția administrare și dezvoltare personal, SA ”Moldtelecom”
2. *Guțu Cătălin*, manager proiecte, ”VICTIANA” SRL.

Adresa Curriculumului în Internet:

https://mecc.gov.md/ro/content/curriculum-invatamintul-profesional-tehnic-postsecundar

Cuprins

I. Preliminarii

Programarea orientată pe obiecte este o paradigma de programare, axată pe ideea încapsulării, adică grupării datelor și codului care operează asupra lor într-o singură structură. Programarea orientată pe obiecte reprezintă unul din cei mai importanți pași făcuți în evoluția limbajelor de programare spre o mai puternică abstractizare în implementarea programelor. Ea a apărut din necesitatea exprimării problemei într-un mod mai natural ființei umane. Astfel, unitățile care alcătuiesc un program se apropie mai mult de modul uman de a gândi.

Statutul Curriculumului. Curriculumul modular "Asistență pentru programarea orientată pe obiecte" este un document normativ şi obligatoriu pentru realizarea procesului de pregătire a tehnicienilor în învățământul profesional tehnic post secundar, care, în conformitate cu sarcinile de lucru, vor care vor acorda asistență în elaborarea şi dezvoltarea diverselor aplicații şi produse-program.

Funcțiile Curriculumului. Funcțiile de bază ale Curriculumului sunt:

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

Beneficiarii Curriculumului. Curriculumul este destinat:

- profesorilor din instituțiile de învăţământ profesional tehnic post secundar;
- autorilor de manuale şi ghiduri metodologice;
- elevilor care își fac studiile la specialitatea în cauză;
- membrilor comisiilor pentru examenele de calificare;
- membrilor comisiilor de identificare, evaluare şi recunoaștere a rezultatelor învățării, dobândite în contexte non-formale şi informale.

Scopul studierii acestui modul constă în formarea şi dezvoltarea competenței profesionale specifice de acordare de asistență în programarea orientată pe obiecte și de organizare a codului programelor la nivel de clase, în mentenanța şi actualizarea produselor-program de sistem precum şi a celui de aplicații. De asemenea, modulul contribuie la dezvoltarea competenței profesionale generale de respectare şi de promovare a normelor de drept informatic.

Unitățile de curs ce în mod obligatoriu trebuie certificate până la demararea procesului de instruire la modulul în cauză:

F.01.O.010 Programarea structurată.
F.02.O.012 Programarea procedurală.
F.03.O.013 Programarea calculatorului.

II. Motivația, utilitatea modulului pentru dezvoltarea profesională

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

- cunoștințe faptice, 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 aplicațiilor.

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

P.04.O.002 Practica de instruire
S.06.O.020 Asistență pentru programarea client-side a site-urilor Web
S.07.O.021 Asistență pentru programarea server-side a site-urilor Web
S.07.A.036 Utilizarea sistemelor de operare pentru dispozitive mobile
S.08.A.038 Programarea aplicațiilor pentru dispozitive mobile

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

III. Competențele profesionale specifice modulului

În cadrul modulului vor fi formate şi dezvoltate următoarele competențe profesionale specifice:

CS1. Organizarea codului de program în stilul orientat pe obiecte.
CS2. Setarea domeniului de valori şi a setului de operații admisibile ale obiectelor.
CS3. Implementarea conceptelor programării orientate pe obiecte în limbaje de programare.
CS4. Modificarea stării şi comportamentului obiectelor în dependență de specificul problemei.
CS5. Acordarea de asistență în elaborarea aplicațiilor orientate pe obiecte.

IV. Administrarea modulului

Semestrul	Numărul de ore				Forma de evaluare	Numărul de credite
	Total	Contact direct		Ore de studiu individual		
		Teorie	Laborator			
IV	90	30	30	30	examen	3

V. Unitățile de învățare

Unități de competență	Unități de conținut	Abilități
1. Inițiere în limbajul orientat spre obiect		
UC1. Cunoașterea tehnologiei de programare ale limbajului orientat spre obiect.	Stiluri de organizarea a codului unui program: - programarea nestructurată; - programarea procedurală; - programarea modulară; - programarea orientată pe obiect. Caracteristici ale limbajului orientat spre obiect și implementarea principiilor de programare orientată spre obiect.	A1. Identificarea tipul de programare structurat, procedural și orientat spre obiect. A2. Selectarea sistemului de operare pentru programare în limbaj. A3. Selectarea mediului de dezvoltare a aplicațiilor și a editoarelor respective.
UC2. Editarea fragmentelor de program.	Vocabularul limbajului. Unități lexicale. Tipurilor de date. Conversii. Structura generală a unui program. Inițializarea și declararea variabilelor Operații de intrare / ieșire la nivel de consolă. Structuri de control.	A4. Identificarea cuvintelor cheie ale limbajului de programare. A5. Inițializarea variabilelor cu valori prestabilite. A6. Utilizarea operatorilor și operanzilor la prelucrarea datelor A7. Realizarea conversiilor de date conform specificațiilor propuse. A8. Identificarea părților componente ale unui program A9. Editarea codului aplicației. A10. Citirea datelor conform specificațiilor propuse. A11. Afișarea datelor conform specificațiilor propuse. A12. Translarea algoritmilor cu structuri de control în limbajul de programare.
2. Clase și obiecte		
UC3. Crearea și instanţierea claselor.	Clase și obiecte. Sintaxă și semantică: - structura unei clase; - modificatori de acces; - date şi metode membre; - constructori, tipuri de constructori;	A13. Prezentarea situațiilor de utilizare a claselor. A14. Crearea unei clase conform specificațiilor propuse. A15. Declararea datelor membre și metodelor membre a clase. A16. Protejarea membrilor clasei conform specificațiilor propuse. A17. Definirea constructorului implicit al unei clase.

Unități de competență	Unități de conținut	Abilități
	- distrugerea obiectelor și eliberarea memoriei; - obiecte; - membri de instanță şi membri de clasă;	A18. Crearea de constructori expliciți conform specificațiilor propuse. A19. Crearea constructorilor de copiere. A20. Inițializarea datelor unui obiect la nivel de constructor. A21. Utilizarea terminologiei specifice conceptelor de încapsulare. A22. Declararea obiectelor. A23. Crearea instanțelor a unei clase. A24. Accesarea datelor și metodelor unui obiect. A25. Inițializarea datelor unui obiect la nivel de metode. A26. Distrugerea obiectelor. A27. Utilizarea claselor interioare. A28. Accesarea datelor din clasele interioare.
UC4. Implementarea modelului conceptual	Crearea modelului conceptual. Limbajul UML - limbaj de modelare unificat. Descrierea limbajului. Tipuri de diagrame. Principii de proiectare.	A29. Reprezentarea claselor prin diagrame. A30. Utilizarea elementelor de bază A31. Elaborarea algoritmilor în conformitate cu diagramele propuse.
3. Conceptele programării orientate spre obiecte		
UC5. Implementarea conceptului de încapsulare	Ascunderea și protecția datelor. Metode accesor. Legătura încapsulării cu descompunerea modulară a unui produs software. Demonstrarea și analiza conceptului de încapsulare în cadrul clasei.	A32. Utilizarea terminologiei specifice conceptelor de încapsulare. A33. Declararea obiectelor. A34. Crearea instanțelor a unei clase. A35. Accesarea datelor unui obiect prin metode accesor. A36. Setarea datelor unui obiect la nivel de metode accesor. A37. Gestionarea obiectelor în cadrul unui program. A38. Elaborarea algoritmilor pentru tipuri de date de tip clasă. A39. Translarea algoritmilor pentru tipuri de date de tip clasă în limbajul de programare.

Unități de competență	Unități de conținut	Abilități
		A40. Implementarea algoritmilor pentru tipuri de date de tip clasă în limbajul de programare.
UC6. Implementarea conceptului de moștenire	Relații între clase: - moștenire; - agregare; - compunere; Clase de bază, clase derivate. Derivarea claselor: - clase de bază; - clase derivate; - protejarea membrilor clasei; - modificatori de acces; - constructori. - destructori Tipuri de moșteniri	A41. Prezentarea situațiilor de aplicare a relațiilor între clase. A42. Reprezentarea mecanismului de moștenire prin diagrama claselor. A43. Prezentarea situațiilor de aplicare a moștenirii. A44. Derivarea unei clase. A45. Derivarea constructorului. A46. Moștenire datelor și metodelor unei clase conform specificațiilor propuse. A47. Derivarea metodelor unei clase. A48. Protejarea datelor între clase. A49. Verificarea corectitudinii definirii structurii claselor ce se află în relații de moștenire. A50. Inițializarea datelor unui obiect prin intermediul constructorului. A51. Crearea ierarhiilor de clase aflate în relația de moștenire conform specificațiilor propuse. A52. Elaborarea algoritmilor pentru clasele de bază/derivate. A53. Translarea algoritmilor clasele de bază/derivate în limbajul de programare. A54. Implementarea algoritmilor pentru clasele de bază/derivate în limbajul de programare. A55. Aplicarea conceptului de moștenire în activitatea profesională.
UC7. Implementarea conceptului de polimorfismului	Polimorfism în cadrul relației de moștenire: - conceptul de polimorfism, - polimorfismul pur; - polimorfism ad hoc; - metode virtuale / abstracte; Clase abstracte / virtuale.	A56. Descrierea conceptului de polimorfism. A57. Prezentarea situațiilor de aplicare a polimorfismului. A58. Redefinirea metodelor claselor în relația de moștenire. A59. Declararea metodelor virtuale. A60. Redefinirea metodelor claselor în relația de moștenire. A61. Declararea claselor virtuale / abstracte.

Unități de competență	Unități de conținut	Abilități
		A62. Prezentarea situațiilor de aplicare a polimorfismului ad-hoc. A63. Elaborarea fragmentelor de program bazate pe clase şi metode virtuale / abstracte. A64. Elaborarea algoritmilor bazați pe conceptul de polimorfism. A65. Translarea algoritmilor bazați pe conceptul de polimorfism în limbajul de programare. A66. Implementarea algoritmilor pe bazați pe conceptul de polimorfism în limbajul de programare. A67. Aplicarea conceptului de polimorfism în activitatea profesională
UC8. Implementarea conceptului de abstractizare	Interfețe - Concepte generale. Definirea unei interfeţe. - Implementarea unei interfeţe. - Interfeţe şi clase abstracte. - Utilitatea interfețelor.	A68. Declararea interfețelor. A69. Apelarea constructorilor în clasele virtuale. A70. Elaborarea fragmentelor de program bazate pe clase şi metode abstracte. A71. Elaborarea algoritmilor bazați pe conceptul de polimorfism. A72. Translarea algoritmilor bazați pe conceptul de polimorfism în limbajul de programare. A73. Implementarea algoritmilor bazați pe conceptul de polimorfism în limbajul de programare. A74. Aplicarea conceptului de polimorfism în activitatea profesională.
4. Structuri dinamice de date		
UC9. Prelucrarea tipurilor dinamice de date la nivel de obiecte	Clase de obiecte. Obiecte de tip listă.	A75. Declararea unei clase de obiecte. A76. Definirea metodelor unei clase de obiecte. A77. Implementarea la nivel de metode a operațiilor specifice unei structuri dinamice de date. A78. Elaborarea algoritmilor pentru tipuri de date dinamice la nivel de obiecte. A79. Translarea algoritmilor pentru obiecte de tip listă. A80. Implementarea algoritmilor pentru obiecte de tip listă în limbajul de programare.

Unități de competență	Unități de conținut	Abilități
UC10. Implementarea funcțiilor și claselor template	Programarea generică. Funcții template. Clase template.	A81. Prezentarea situațiilor de aplicare a programării generice. A82. Elaborarea funcțiilor template. A83. Utilizarea claselor template conform specificațiilor propuse. A84. Elaborarea algoritmilor pentru funcții și clase template. A85. Translarea algoritmilor pentru funcții și clase template. A86. Implementarea algoritmilor pentru funcții și clase template în limbajul de programare.
UC11. Implementarea conceptului de agregare	Ierarhii de clase.	A87. Reprezentarea agregării în diagrama claselor. A88. Crearea ierarhiilor de clase formate din 3-5 clase. A89. Elaborarea algoritmilor pentru ierarhii de clase. A90. Translarea algoritmilor pentru ierarhii de clase. A91. Implementarea algoritmilor pentru ierarhii de clase în limbajul de programare. A92. Aplicarea conceptelor: abstractizare, incapsulare, moștenire, polimorfism și agregare în activitatea profesională.
UC12. Îndepărtarea erorilor	Tratarea excepțiilor 1. Ierarhia claselor ce descriu excepții. Instrucțiunile *try-catch*, *try-catch-finally*, *throw*. 2. Avantaje privind tratarea excepțiilor. Definirea de excepții utilizator.	A93. Tratare a excepțiilor. A94. Aruncarea excepțiilor. A95. Crearea propriilor excepții.

VI. Repartizarea orientativă a orelor pe unități de învățare

Nr. crt.	Unități de învățare	Numărul de ore			
		Total	Contact direct		Ore de studiu individual
			Teorie	Laborator	
1.	Inițiere în limbajul orientat spre obiect	6	2	4	0
2.	Clase și obiecte	30	10	10	10
3.	Conceptele programării orientate spre obiecte	28	10	8	10
4	Structuri dinamice de date	26	8	8	10
	Total	**90**	**30**	**30**	**30**

VII. Studiu individual ghidat de profesor

Materii pentru studiul individual	Produse de elaborat	Forma de evaluare	Termeni de realizare
1. Concepte ale programării orientată pe obiecte			
Clase și obiecte	Proiect individual: Produse program cu utilizarea claselor și obiectelor.	Prezentarea proiectului elaborat. Argumentarea orală.	Săptămâna 7
Conceptele programării orientate spre obiecte	Proiect individual: Produse program cu utilizarea ierarhiilor de clase.	Prezentarea proiectului elaborat. Argumentarea orală.	Săptămâna 14

VIII. Lucrările practice recomandate

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

1. Aplicarea tipurilor de date de bază la elaborarea programelor.
2. Clase și obiecte. Modificatori de acces.
3. Utilizarea constructorilor și destructorului.
4. Prelucrarea tipurilor de date frecvent utilizate (structuri, masive etc) la nivel de clase.
5. Implementarea relației de moștenire simplă.
6. Implementarea diverselor tipuri moșteniri.
7. Implementarea polimorfismul dinamic.
8. Implementarea polimorfismul ad-hoc.
9. Implementarea ierarhiilor formate din 3-5 clase.

10. Produse program cu utilizarea claselor imbricate.
11. Produse program cu utilizarea diverselor tipuri de moșteniri prin intermediul interfețelor.
12. Prelucrarea tipurilor de date dinamice la nivel de clase.
13. Prelucrarea datelor la nivel funcții și clase template.
14. Dezvoltare de aplicații cu utilizarea structurilor dinamice de date.
15. Dezvoltare de aplicații îndepărtând erorile.

IX. Sugestii metodologice

Elementul de bază al Curriculumului sunt competențele ce trebuie formate şi 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-şi 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 şi acțiuni prestabilite.

Activitățile pot fi organizate cât cu prezența fizică, atât și **online**. Pentru a crea un mediu de învățare cât mai aproape de o clasă reală, este suficient de utilizat trei tipuri de resurse:

- O platformă de interacțiune în timp real, cu video și text, cu elevii printre care sunt: Google Meet, Jitsi Meet, Zoom, Discord, Cisco Webex, Microsoft Teams etc.
- Aplicații sau platforme de colaborare online, care facilitează schimbul de documente, teste sau teme pentru acasă, între profesori și elevi și înregistrează o evidență a acestora, care permite și feedback din partea profesorului. Aici recomandăm Moodle, Google Classroom și alternative folosite de profesori, precum Edmodo, EasyClass și ClassDojo.
- Resurse și aplicații de învățare pe care le poate crea profesorul sau resurse deja existente sub formă de prezentări, lecții, fișe, imagini și clipuri care pot fi folosite atât în timpul lecțiilor live, cât și ca teme de lucru pentru acasă. Aici lista e mai lungă și include aplicații precum TestMoz, LearningApps, Mentimetr, LinoIt, ASQ, Kahoot, Quizziz, Wordwall, Padlet, Twinkl sau Digitaliada, precum și surse de inspirație pentru filme, teme și studiu individual.

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

Simularea şi 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-evaluare următoarelor materii Sintaxa şi semantica claselor, moștenire, ierarhii de clase ș. a.

Problematizarea mai poate fi denumită şi predare prin rezolvare de probleme sau predare productivă de probleme. Conform acestei metode instruitului este pus în fața unor dificultăți create în mod deliberat, şi 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 situației de tip problemă se va ține cont de următoarele caracteristici:

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

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

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

Această metodă se recomandă pentru predarea-învățarea-evaluare următoarelor materii constructori, moștenire simplă/multiplă, polimorfismul dinamic ş. a.

Algoritmizarea reprezintă o metodă de predare-învățare bazată pe utilizarea şi valorificarea algoritmilor în procesul de instruire. Algoritmul de instruire se reprezintă sub forma unui grup de scheme, unui set de operații, iar prin parcurgerea lor într-o ordine bine stabilită duce 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-evaluare următoarelor materii polimorfismul ad-hoc, programarea generică, ierarhii de clase ş. a.

Metoda studiul de caz valorifică o situației reală care se analizează şi 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 şi rezolvarea problemei. În utilizarea acestei metode se conturează câteva etape: 1) Selectarea şi prezentarea cazului; 2) Organizarea echipelor de lucru; 3) Prelucrarea şi conceptualizarea; 4) Structurarea finală a studiului. Această metodă se recomandă pentru predarea-învățarea-evaluare următoarelor materii ierarhii de clase, polimorfismul dinamic.

X. Sugestii de evaluare a competențelor profesionale

Evaluarea competențelor profesionale este procesul prin care sunt colectate şi 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 şi colaborarea dintre

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

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

Activitățile de evaluare vor fi orientate spre motivarea elevilor şi obținerea unui feedback continuu, fapt ce va permite corectarea operativă a procesului de învățare, stimularea autoevaluării şi 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 cunoștința elevilor tematica lucrărilor, modul de evaluare (bareme/grile/criterii de notare) şi condițiile de realizare a fiecărei evaluări.

Evaluarea inițială (datorită funcțiilor ei de diagnoză, respectiv prognoză), are rolul de a stabili care este nivelul de pregătire al elevilor la începutul unei activități, pentru a putea adopta o strategie, o tehnologie didactică care să corespundă realităților.

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 şi abilități. În acest scop, vor fi clar stabiliți indicatorii şi descriptorii de performanță ai procesului şi 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 şi 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.;
- ***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 prin care elevul își va demonstra abilitățile formate.

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

- implementarea conceptelor programării orientate pe obiecte;
- crearea structurilor de clase;
- organizarea codului de program în stilul orientat pe obiecte;
- reprezentarea conceptelor programării orientate pe obiecte prin diagrama claselor;
- modificarea stării și comportamentului obiectelor 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;
- obiecte gestionate conform specificațiilor propuse.

Criteriile de evaluare a produselor pentru măsurarea competenței vor include:

- Utilizarea corectă a mecanismelor programării orientate pe obiecte.
- Corectitudinea claselor elaborate.
- Fundamentarea deciziilor.
- Ținuta lingvistică.
- Respectarea termenilor de elaborare.

Forma finală de evaluare a modulului este examenul. Pentru a fi admis la examen elevul trebuie să susțină evaluările curente, să însușească materialul teoretic și să îndeplinească sarcinile propuse

pentru 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

Cerinţe față de sălile de curs	
Pentru orele teoretice	Cabinet de informatică cu 15 calculatoare Proiector
Pentru orele de laborator	Laborator de informatică care asigură fiecărui elev un calculator
Cerinţe pentru lecții online	
Pentru orele teoretice și de laborator	Instrumente de comunicare sincronă Platforme de colaborare online Instrumente digitale de evaluare
Cerinţe 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 Code::Blocks Dev C/CPP Visual Studio 2019

XII. Resursele didactice recomandate elevilor

Nr. crt.	**Denumirea resursei**	**Locul în care poate fi consultată/ accesată/ procurată resursa**
1.	Atanasiu Irina, Costinescu B., Dragoi O.A., Popovici F.I. Limbajul Java. O perspectiva pragmatică. Editura Agora, Tg. Mureș, 2004.	Biblioteca instituției
2.	Prodan A., Prodan M. *Mediul Java pentru Internet.* Editura Promedia Plus, Cluj-Napoca,1997.	Biblioteca instituției
3.	Lemay L., Cadenhead R. Java2 Fără profesor în 21 zile. Editura Teora, București, 2009.	Biblioteca instituției
4.	Chan M.C., Griffith S.W., Iasi A.F. *Java – 1001 secrete pentru programatori.* Editura Teora, București, 2006.	Biblioteca instituției
5.	B. Eckel, Thinking in Java, Prentice Hall (4-th Edition), 2006.	Biblioteca instituției

Nr. crt.	Denumirea resursei	Locul în care poate fi consultată/ accesată/ procurată resursa
6.	C. Olaru, S. Tanasa, Java de la 0 la expert, Polirom, Iasi, 2003	Biblioteca instituției
7.	C. Frăsinaru, Curs practic de Java, Bucuresti, Ed. Matrix Rom	Biblioteca instituției
8.	G. Stoian, C.I. Popirlan, Tehnologii Java pentru dezvoltarea aplicatiilor - Note de curs, Seria Computer Science, Editura Universitaria, Craiova, 2009.	Biblioteca instituției
9.	D.Danciu, G.Mardale - Arta programării în JAVA (Vol. I şi II), Editura Albastra 2004.	Biblioteca instituției
11.	Curs Java.	http://java.sun.com
14.	A. Braicov, S. Gîncu, Borland C++ Builder. Chişinău, 2009.	http://en.calameo.com/read/00280 1569838bdc5a9164
15.	A. Ruceanu, Programarea orientată pe obiecte: limbajul C++, 2007.	http://www.runceanu.ro/adrian/wp -content/cursuri/poo2013.php
16.	I. Lupu, V. Cabac, S. Gîncu Formarea şi dezvoltarea competenței de programare orientată pe obiecte la viitorii profesori de informatică, Chișinău, 2013.	http://en.calameo.com/books/0028 015690bbcfd6a9e03
17.	S. Gîncu Metodologia rezolvării problemelor de informatică în stilul orientat pe obiecte, Chişinău, 2012.	http://en.calameo.com/read/00280 1569ce96f41ef59f
18.	Programarea orientată pe obiect și programarea vizuală.	http://colegiulbratianu.ro/wp-content/themes/theme53309/documente/software/DotNet.pdf
19.	Booch G. Object-Oriented Design with Applications. Benjamin/Cummings, Redwood City, California, 2nd edition, 1994.	http://dl.acm.org/citation.cfm?id=1 74890
20.	Фаулер М. UML. Основы / Пер. с англ. 3-е изд. СПб: Символ-Плюс, 2005.	http://213.230.96.51:8090/files/ebooks/dasturlash/Uml%20osnovy%203-e%20izd.pdf
21.	Reprezentarea UML al claselor	http://inf.ucv.ro/~mihaiug/courses/poo/slides/Curs%2005%20-%20UML.pdf