

**Ministerul Educației al Republicii Moldova
Universitatea de Stat „Alec Russo” din Bălți
Facultatea de Științe Reale, Economice și ale Mediului
Catedra de matematică și informatică**

Curriculum

la unitatea de curs

Programare funcțională (Python, Scala, etc.)

Ciclul I, studii superioare de licență

Codul și denumirea domeniului general de studiu: 44 Științe exacte

Codul și denumirea specialității: 444.1 Informatica

Forma de învățământ: cu frecvență redusă

Autor: Mircea PETIC,
dr., conf. univ.

Bălți, 2017

Discutat și aprobat la ședința Catedrei de matematică și informatică
Procesul-verbal nr. 15 din 09.06.2017

Șeful Catedrei de matematică și informatică

Echevici conf. univ., dr. Eugeniu PLOHOTNIUC

Discutat și aprobat la ședința Consiliului Facultății de Științe Reale,
Economice și ale Mediului

Procesul-verbal nr. 15 din 27.06.2017

Decanul Facultății de Științe Reale, Economice și ale Mediului

Ciobanu conf. univ. dr. Ina CIOBANU



Informații de identificare a unității de curs

Facultatea: Științe Reale, Economice și ale Mediului

Catedra de matematică și informatică

Domeniului general de studii: 44 Științe exacte

Domeniul de formare profesională: 444 Informatica

Denumirea specialității: 444.1 Informatica

Denumirea unității de curs: Programarea funcțională (Python, Scala, etc.)

Administrarea unității de curs

Codul unității de curs	Credite ECTS	Total ore	Repartizarea orelor				Forma de evaluare	Limba de predare
			Prelegeri	Seminar	Laborator	Lucrul indiv.		
S.04.A.121	5	150	12	-	18	120	Examen	Română

Anul de studii și semestrul în care se studiază: anul II, sem. III

Regimul unității de curs: obligatorie (la libera alegere)

Categoria formativă: unitate de curs de specializare.

Informații referitoare la cadrele didactice



Mircea Petic, dr. în informatică, conferențiar universitar. Absolvent al Universității de Stat „Alec Russo” din Bălți, specialitatea „Informatica și limba engleză aplicată”. A efectuat stagii în diverse universități, inclusiv peste hotare, unde s-a specializat în domeniul informaticii, utilizarea tehnologiei informației și a comunicațiilor în învățământ, procesarea limbajului natural, programarea obiect orientată, programarea logică și funcțională, ingineria

programării, managementul proiectelor.

E-mail: petic.mircea@yahoo.com

Orele de consultații - miercuri: 15.00 -17.00. Consultațiile se oferă atât în regim „față-în-față”, cât și prin utilizarea poștei electronice, YahooMessenger (petic.mircea) și Skype sau rețeaua socială facebook. Numele în Skype – mir_cescu.

Descrierea unității de curs

Unitatea de curs „*Programarea funcțională (Python, Scala, etc.)*” este un curs facultativ care permite aprofundarea cunoștințelor în programare. Realizarea unui sistem informatic reprezintă o activitate complexă și de durată, ce antrenează mari resurse materiale, umane și de timp. Programarea funcțională este o paradigmă de programare care tratează calculul ca evaluare de funcții matematice și evită starea și datele muabile. Se pune accent pe aplicarea de funcții, spre deosebire de programarea imperativă, care folosește în principal de schimbările de stare. Modelul matematic al programării funcționale îl reprezintă calculul lambda. Limbajele funcționale moderne pot fi considerate extensii ale calculului lambda. Noțiunea de bază în această paradigmă este cea de funcțională sau funcție de nivel înalt, o funcție care poate accepta ca argument sau returna ca valoare o altă funcție.

Limbajele de programare funcționale, mai ales cele pur funcționale, sunt promovate mai ales în mediile academice, fiind rar folosite în dezvoltarea de software comercial. În cursul dat vor fi studiate aspecte programării funcționale în baza limbajelor Scala, Python, Lisp și Haskell.

Integrarea în programul de studii

Studierea unității de curs „*Programarea funcțională (Python, Scala, etc.)*” se sprijină pe cunoștințele, capacitățile și competențele dezvoltate în cadrul unităților de curs „*Bazele programării*” și „*Programarea structurată*” studiate în ciclul I de licență. Finalitățile și conținutul unității de curs sînt corelate cu finalitățile și conținuturile unităților de curs menționate mai sus.

Competențe prealabile

1. *Bazele programării*: Tipuri de date simple și structurate, expresii.
2. *Programarea structurată*: Instrucțiunile simple și compuse (condiționale și repetitive). Funcții. Proceduri. Recursivitate.

Competențele dezvoltate în cadrul unității de curs

Competențe profesionale:

CP1. Operarea cu fundamentele științifice ale informaticii și matematicii și utilizarea acestor noțiuni în comunicarea profesională.

CP2. Elaborarea modelelor pentru descrierea fenomenelor și proceselor reale.

CP3. Proiectarea, elaborarea și analiza algoritmilor pentru rezolvarea problemelor.

CP4. Programarea, dezvoltarea și mentenanța aplicațiilor informatice în limbaje de nivel înalt.

Competențe transversale:

CT1. Aplicarea regulilor de muncă riguroasă și eficientă, manifestarea unei atitudini responsabile față de domeniul profesional, pentru valorificarea optimă și creativă a propriului potențial în situații specifice, cu respectarea principiilor și a normelor de etică profesională.

CT3. Identificarea oportunităților de formare continuă și valorificarea eficientă a resurselor și tehnicilor de învățare pentru propria dezvoltare.

Finalitățile unității de curs

La finalizarea studierii unității de curs studentul va fi capabil să

1. definească conceptul programare funcțională;
2. cunoască destinația și modul de lucru cu funcționale;
3. posede deprinderi de lucru cu expresiile lambda;
4. utilizeze corect evaluarea strictă și non-strictă;
5. posede deprinderi de lucru cu evaluarea lazy;
6. explice corect modul de programare funcțional cu limbaje nefuncționale;
7. explice corect diferența între programarea funcțională și imperativă;

Conținuturi

a) Prelegeri

Nr. d/o	Conținuturi	Nr. de ore
1	Programare și limbaje de programare. Importanța programării funcționale ca nouă metodologie de programare. Medii de dezvoltare Scala. Funcții de nivel înalt. Structuri de date funcționale. Gestionarea erorilor fără excepții. Funcții stricte și non-stricte.	2
2	Paralelism pur funcțional. Testarea bazată pe proprietăți. Parsări. Monoizi. Monazi. Functori aplicativi și transversabili. Lucrul cu datele externe.	2
3	Introducere în limbajul de programare funcțională Haskell. Deosebiri și asemănări cu Scala. Studiu de caz în limbajul de programare funcțională Haskell.	2
4	Deosebiri și asemănări între limbajele de programare Scala și CLISP. Studiu de caz în limbajul de programare funcțională CLISP.	2
5	Programarea funcțională în limbaje nefuncționale (Python). Programarea funcțională în limbaje nefuncționale (Java și C#)	2
6	Lucrarea de control	2
Total		12

b) *Laborator*

Nr. d/o	Tematica lecțiilor de laborator	Nr. de ore
1	Medii de programare funcțională pe Scala. Module. Obiecte și spații de nume.	2
2	Funcții de nivel înalt. Funcții polimorfice. Structuri de date funcționale.	2
3	Sarcini individuale nr. 1	2
4	Gestionarea erorilor fără excepții. Funcții stricte și non-stricte. Paralelism funcțional.	2
5	Parsarea documentelor JSON. Monoizi. Monazi. Lucrul cu datele externe.	2
6	Sarcini individuale nr. 2	2
7	Programarea funcțională în limbajul Python	2
8	Sarcini individuale nr. 2	2
9	Susținerea proiectului Scala	2
Total		18

Activități de lucru individual

Proiectul va presupune elaborarea unei aplicații Scala care rezolvă o problemă din viața reală. Aplicația va conține și o descriere a problemei soluționate și a modului în care a fost rezolvată. Codul aplicației va avea comentarii explicative. Descrierea aplicației va fi prezentată într-un raport editat într-un document Word pe 3-4 pagini format A4, font #12, 1,5 intervale. Activitatea va fi evaluată atât de către colegi cât și de către titularul disciplinei într-o ședință aparte.

Criterii de evaluare:

- Corectitudinea rezolvării problemei prin elaborarea aplicației Scala;
- Relevanța și valoarea comentariilor;
- Exactitate (logică, ortografică) a raportului prezentat;

Termenul limită (deadline) de prezentare a sarcinii – perechea a 9-a (laborator)

Exemple de teme pentru proiect

1. Numărarea cuvintelor dintr-un fișier txt scris în limba română cu diacritice (cel puțin 1000 cuvinte). Separatori de cuvinte sînt: spațiile libere, semnele de punctuație, simbolul ”-”.
2. Este nevoie de a automatiza evidența iepurilor din crescătorie. Se cunosc cîți iepuri sînt la începutul fiecărei luni, cîți au murit și cîți s-au născut în cursul fiecărei luni.
3. Determinarea greutateii ideale a unei persoane cunoscînd înălțimea, vîrsta și sexul persoanei. Formulele de calcul sînt: $G_{\text{masculin}} = 50 + 0.75 * (\text{înălțime} - 150) + (\text{vîrsta} - 20)/4$, $G_{\text{feminin}} = G_{\text{masculin}} - 10$, unde înălțimea este exprimată în cm și vîrsta în ani. Sexul indică separat.
4. O bază de date conține un tabel cu cîteva cîmpuri. Se cere să se transforme acest tabel într-un fișier JSON sau fișier txt. Numele bazei de date și cel al fișierelor de ieșire se va indica prin interfață.
5. Este nevoie de a transforma o suma de bani dintr-o valută în alta. În acest scop se va utiliza cursul de cumparare și cel de vînzare pentru valutele corespunzătoare. Se lucrează cu valutele: leul moldovenesc, dolarul american, euro, leul românesc, rubla rusă și hrivna ucraineană.

Resursele informaționale la unitatea de curs

A. Literatura de bază

1. CHIUSANO, P. BJARNASON, R. *Functional Programing in Scala*, Manning Shelter Island, USA, 322 p.
2. ДУШКИН, Р. *Функциональное программирование на языке Haskell*, ДМК, Москва, 2016, 608 стр.

B. Literatura suplimentară

3. ХОРСТМАН, К. *SCALA для нетерпеливых*, ДМК, Москва, 2015, 408 стр.
4. ДУШКИН, Р. В. *Практика работы на языке Haskell*, ДМК, Москва, 2010, 288 стр.
5. РОГАНОВА, Н. *Функциональное программирование*, МГИУ, Москва, 2010, 215 стр.
6. MERTZ, D. *Functional programming with Python*, O'REILLY Media Inc., 2015, 49 p.

C. Resurse Internet

7. Situl oficial IDE IntelliJIDEA [online], [vizitat 09.10.2016] Disponibil: < www.jetbrains.com >
8. Situl oficial Scala IDE [online], [vizitat 09.10.2016] Disponibil: <<http://scala-ide.org/>>
9. Online compiler and IDE [online], [vizitat 09.10.2016] Disponibil: <<http://ideone.com/>>
10. Documentația oficială Scala [online], [vizitat 09.10.2016] Disponibil: < <http://www.scala-lang.org/> >

Evaluarea

Cunoștințele, capacitățile și competențele studenților vor fi evaluate:

1. La prelegeri (**PR**):
 - 1.1. *Lucrarea de control*: perechea a 6-a (**LC1**).
2. În cadrul lecțiilor de laborator (**LLab**):
 - 2.1. *Sarcini individuale nr. 1*: perechea a 3-a (**SI1**)
 - 2.2. *Sarcini individuale nr. 2*: perechea a 6-a (**SI2**)
 - 2.3. *Sarcini individuale nr. 3*: perechea a 8-a (**SI2**)
 - 2.4. *Proiect*: perechea a 9-a (**Pro**)
3. La examenul final, conform orarului întocmit de decanat (**Ex**).

Nota finală la disciplina „Programarea funcțională” se calculează conform formulelor:

$$N_{\text{evaluarea curentă}} = 1/2 \times \text{PR} + 1/2 \times \text{LLab}$$

$$N_{\text{finală}} = 0,5 \times N_{\text{evaluarea curentă}} + 0,5 \times N_{\text{examen}},$$

Unde **PR=LC1** și **LLab=(SI1+SI2+SI3+Pro)/4**

Examenul final se susține scris, care va include un test complex cu diferite tipuri de itemi. Pentru a fi admis la examen, este obligator ca ambele note (**PR** și **LLab**) să fie cel puțin 5.0.

Recuperarea notelor și susținerea repetată a examenului are loc în datele stabilite de orarul întocmit de decanat de susținere a restanțelor.

Baremul de apreciere

(în baza REGULAMENTULUI

cu privire la evaluarea rezultatelor academice ale studenților în
Universitatea de Stat „Alec Russo” din Bălți)

Procentajul	Nota
100 – 91	10

90 – 81	9
80 – 71	8
70 – 66	7
65 – 61	6
60 – 51	5
50 – 41	4
40 – 31	3
30 – 16	2
15 – 0	1

Principiile de lucru în cadrul disciplinei

1. Calendarul cursului (termenii-limită de prezentare a sarcinilor propuse spre rezolvare, momentele de evaluare etc.) este corelat cu calendarele la alte discipline din semestru. De aceea prezentarea sarcinilor după termenul-limită indicat în calendar nu este salutăată, iar studenții care amână frecvent prezentarea sarcinilor își formează o imagine nefavorabilă.
2. Nu este salutăată întârzierea la ore.
3. Este salutăată poziția activă a studentului, care studiază din propria inițiativă noi conținuturi, propune soluții (aplicații, instrumente Web), formulează întrebări în cadrul prelegerilor și a orelor practice.
4. În cadrul disciplinei o atenție sporită va fi oferită respectării principiilor *etice*. Prezentarea unor soluții a sarcinilor, preluate de la colegi sau din alte surse, preluarea informațiilor din diverse surse, fără a face trimitere la sursă, va fi considerată *plagiat* și va fi sancționată prin note de „1” .