RTU studiju kurss "Objektorientētā programmēšana"

33000 Datorzinātnes, informācijas tehnoloģijas un enerģētikas fakultāte

Vispārējā informācija

Kods	DE0571
Nosaukums	Objektorientētā programmēšana
Studiju kursa statuss programmā	Obligātais/Ierobežotās izvēles
Atbildīgais mācībspēks	Gundars Alksnis - Doktors, Asociētais profesors
Apjoms daļās un kredītpunktos	1 daļa, 6.0 kredītpunkti
Studiju kursa īstenošanas valodas	LV, EN
Anotācija	Studiju kursā tiek aplūkoti pamata un vidēja līmeņa objektorientētās programmēšanas (OOP) temati, kā primāro demonstrēšanas programmēšanas valodu izmantojot C++. Veiksmīgi pabeidzot studiju kursu, studentiem jāizprot OOP koncepcijas un jābūt kompetentiem no projektējuma uzrakstīt objektorientēta risinājuma pirmkodu. Tiek apskatītas tādas OOP pamatkoncepcijas, kā abstrakcija, iekapsulēšana, mantošana, polimorfisms un modularitāte. Papildus tiek izskatīt arī temati, kas saistīti ar funkciju pārlādi, funkciju pārdefinēšanu, izņēmumu apstrādi, šabloniem un klašu bibliotēku izmantošanu. Izskatītie temati tiek salīdzināti ar citām OOP valodām, piemēram, Java un C#, uzsverot, ka svarīgāk par konkrētas programmēšanas valodas sintaksi ir izprast OOP principus. Būtisku studiju kursa daļu veido praktisku uzdevumu izpilde, kuros studenti pielieto iegūtās zināšanas un nostiprina OOP prasmes.
Mērķis un uzdevumi, izteikti kompetencēs un prasmēs	Studiju kursa mērķis ir apmācīt un virzīt studentus OOP pamatkoncepciju apguvē, lai studiju kursa nobeigumā viņi būtu kompetenti tos pielietot programmatūras izstrādē un uzturēšanā. Studiju kursa uzdevumi: - izskaidrot OOP pamatkoncepcijas (abstrakcija, iekapsulēšana, mantošana, modularitāte, polimorfisms); - nostiprināt prasmes lasīt iepriekš uzrakstītu pirmkodu; - attīstīt prasmes no sagatavota projektējuma uzrakstīt OOP pirmkodu, pielietojot klašu bibliotēku datu tipus un funkcionalitāti; - nostiprināt programmatūras izstrādes un atkļūdošanas rīku lietošanas prasmes.
Patstāvīgais darbs, tā organizācija un uzdevumi	Studentiem jāizpilda vairāki praktiski uzdevumi, kas ietver teorētisko sagatavošanos, risinājuma programmēšanu un rakstisku secinājumu izdarīšanu. Uzdevumi ir sadalīti atbilstoši tematiskām grupām: - ievads integrētā izstrādes vidē; - abstrakcija un iekapsulēšana; - pārskaitāmie datu tipi, objektu atsauces un kvalifikatori; - klašu konstruktori, inicializācijas saraksti, funkciju pārlāde, klases atbildība; - klašu atkarības, funkcionalitātes deleģēšana, objektu rādītāji; - operatoru pārlāde, ievades/izvades apstrāde, izņēmumu apstrāde; - mantošana un dinamiskais polimorfisms; - klašu bibliotēkas datu tipi darbam ar konteineriem un algoritmiem.
Literatūra	Obligātā / Obligatory: 1. C++ Get Started! / Internet: https://isocpp.org/get-started 2. Bjarne Stroustrup. A Tour of C++ Addison-Wesley, 2014 Papildu / Additional: 1. Ray Lischner. Exploring C++20, Apress, 2020 2. J. Burton Browning, Bruce Sutherland, "C++20 Recipes", Apress, 2020 3. Scott Meyers. Effective Modern C++ O'Reilly Media, 2014 4. Bjarne Stroustrup. The C++ Programming Language 4th Edition Addison-Wesley, 2013 5. Grady Booch, etc. Object-Oriented Analysis and Design with Applications 3rd Ed. Addison-Wesley, 2007 Citi / Other: 1. RTU E-library's ProQuest Ebook Central (keywords: C++; object-oriented programming) 2. Online Videos, e.g.: https://www.youtube.com/playlist?list=PLlrATfBNZ98dudnM48yfGUldqGD0S4FFb, https://www.youtube.com/playlist?list=PL43pGnjiVwgQHLPnuH9ch-LhZdwckM8Tq
Nepieciešamās priekšzināšanas	Zināšanas par strukturālo programmēšanu (vēlams C saimes programmēšanas valodā).

Studiju kursa saturs

Saturs	Pilna un nepilna laika klātienes studijas		Nepilna laika neklātienes studijas	
	Kontakt stundas	Patstāv. darbs	Kontakt stundas	Patstāv. darbs
Studiju kursa organizācija un vērtēšana. Mācību materiāli. Programmēšanas valodu attīstības tendences. Programmatūras izstrādes dzīves cikls un studiju kursa loma tajā.	2	2	0	0
C++ programmas pirmkoda analīze. Kompilēšanas un saistīšanas procesi.	2	2	0	0
Programmēšanas paradigmas. C++ vēsture un filozofija. Abstrakcija un iekapsulēšana. Objekti un klases. Klasifikācijas problēmas.	2	2	0	0
Klases elementu pieejas modifikatori. Iebūvētie un atvasinātie datu tipi. C++ pirmsprocesors.	2	2	0	0
Deklarācija un definīcija. Objekti, atsauces un rādītāji. Rakstzīmju virknes. Pārskaitāmie datu tipi.	2	2	0	0
Datu tipu pārveidošana. Datu masīvi. Redzamības apgabali. Strukturālās programmēšanas aspekti.	4	4	0	0

Objektu glabāšanas datora atmiņā. Funkciju pārlāde jeb statiskais polimorfisms. Konstruktori. Destruktori.	4	4	0	0
Konstruktoru un destruktoru darbība funkcijās. Dinamiskā atmiņa. C++ viedie rādītāji. 1. kontroldarbs.	4	4	0	0
1. kontroldarba analīze. Klases String realizācija. Klases statiskie elementi.	4	4	0	0
C++ klases draugi. Operatoru pārlāde.	2	2	0	0
Standartizētu klašu bibliotēku apskats. Darbs ar ievades/izvades apstrādes klasēm.	4	4	0	0
Izņēmumu apstrāde. 2. kontroldarbs.	4	4	0	0
2. kontroldarba analīze. Ievads vienotajā modelēšanas valodā UML. Asociācijas un daudzkārtības starp klasēm. Mantošana un datu tipu atvasināšana.	4	4	0	0
Ievads SOLID principos. Izvēle starp mantošanu un kompozīciju. Klases virtuālās funkcijas un destruktori. Klases tīri virtuālās jeb abstraktās funkcijas. Abstraktas klases un interfeisi.	4	4	0	0
Šabloni un vispārīgie datu tipi.	4	4	0	0
Objektorientētās projektēšanas principi. Standarta klašu bibliotēkas konteineri un algoritmi. 3. kontroldarbs.	4	4	0	0
3. kontroldarba analīze. OOP valodu salīdzinājums: C++, Java un C#.	8	8	0	0
Programmēšanas uzdevumi	16	16	0	0
Eksāmens	4	4	0	0
Kopā:	80	80	0	0

Sasniedzamie studiju rezultāti un to vērtēšana

Sasniedzamie studiju rezultāti	Rezultātu vērtēšanas metodes
Spēj izskaidrot OOP pamatkoncepciju – abstrakcijas, iekapsulēšanas, mantošanas, modularitātes un polimorfisma – būtību un pielietošanu programmēšanas valodā C++.	Eksāmens un praktiskie uzdevumi. Kritēriji: spēj atpazīt pamatkoncepciju realizācijas atšķirības citās OOP valodās.
Spēj saprast trešo pušu sagatavotu pirmkodu un izskaidrot tajā izmantoto priekšrakstu loģiku.	Praktiskie uzdevumi. Kritēriji: spēj konstatēt nepilnības pirmkodā un zina, kā tās novērst.
Spēj uzrakstīt kādas abstrakcijas deklarāciju un definīciju klases datu tipā.	Praktiskie uzdevumi. Kritēriji: spēj sadalīt atbildības starp vairākām klasēm.
Spēj izskaidrot objektu redzamības apgabalus un objektu glabāšanai var pielietot lokālo, statisko, dinamisko un globālo atmiņas veidu.	Eksāmens un praktiskie uzdevumi. Kritēriji: spēj novērtēt atbilstošāko atmiņas veidu izvēli konkrētās situācijās.
Spēj izmantot gatavas klases no standarta klašu bibliotēkas darbam ar rakstzīmju virknēm, ievadi/izvadi, failiem un spēj lietot izņēmumu apstrādes mehānismu.	Praktiskie uzdevumi. Kritēriji: pārzina atšķirības starp līdzīgiem datu tipiem, spēj pielietot vairākus datu tipus, lai panāktu vēlamo risinājumu.
Spēj izmantot datu tipu šablonus.	Eksāmens un praktiskie uzdevumi. Kritēriji: izprot, kad jālieto datu tipi un kad to šabloni.
Pārzina objektorientētas projektēšanas principus un, kā to pielietošana palīdz izstrādāt un uzturēt sarežģītu programmatūru.	Praktiskie uzdevumi. Kritēriji: prot izvērtēt SOLID principu nepieciešamību.
Spēj salīdzināt dažādu OOP valodu iespējas.	Eksāmens un praktiskie uzdevumi. Kritēriji: prot salīdzināt programmēšanas valodas C++, Java un C#.

Studiju rezultātu vērtēšanas kritēriji

Kritērijs	% no kopējā vērtējuma
Praktiskie uzdevumi	40
Eksāmens	50
Bonuss par sekmīgi uzrakstītiem diviem kontroldarbiem un iesniegtiem risinājumiem visiem praktiskiem uzdevumiem	10
Kopā:	100

Studiju kursa plānojums

Daļa	KP	Stundas			Pārbaudījumi		
		Lekcijas	Prakt d.	Laborat	Ieskaite	Eksām.	Darbs
1.	6.0	40.0	0.0	40.0		*	