KATALOG ZNANJA

1. IME PREDMETA

PROGRAMIRANJE I
PROGRAMIRANJE II

2. SPLOŠNI CILJI

Splošni cilji predmeta so:

- usposabljanje za samostojno reševanje problemov, ki jih lahko rešimo z računalnikom,
- razvijanje sposobnosti sprejemanja in upoštevanja sodelujočih v razvoju programske opreme,
- spoznavanje značilnosti dela v skupini (organiziranje, vodenje, analiziranje dela v skupini),
- oblikovanje navad za redno spremljanje razvoja na področju programskih razvojnih orodij,
- kritično vrednotenje svojega dela.

3. PREDMETNO-SPECIFIČNE KOMPETENCE

Pri predmetu si študenti poleg generičnih pridobijo naslednje kompetence:

- izdelajo analizo zahtev in načrt razvoja programske opreme na podlagi zahtev naročnika programske opreme,
- napišejo preproste aplikacije,
- izdelajo konzolni, grafični in spletni vmesnik aplikacije,
- testirajo programsko opremo,
- dokumentirajo programsko opremo,
- opišejo uporabnost programskega izdelka,
- določajo ekonomsko vrednost konkretne programske rešitve,
- implementirajo programske rešitve v nova okolja.

4. OPERATIVNI CILJI

INFORMATIVNI CILJI	**FORMATIVNI CILJI**
Študent:	Študent:
PROGRAMIRANJE I	
1. UVOD V PROGRAMSKE JEZIKE	
• pozna faze življenjskega cikla programske opreme, • opiše vlogo programskih jezikov, • našteje potrebno programsko opremo za razvoj aplikacij, • pozna avtorske pravice pri pisanju in uporabi programske opreme, • spozna razvojno okolje.	• izvede predhodno raziskovanje in analizo problema, ki ga rešuje, • našteje nekatere delitve programskih jezikov, • namesti prevajalnik in razvojno okolje, • napisan program zna prevesti in zagnati, • preveden program zna prenesti na drug sistem z istim ali drugim operacijskim sistemom, • pozna potrebne licence za razvojno okolje, ki ga uporablja za razvoj programske opreme, • uporablja razvojno okolje za pisanje programske opreme; program napisan in preveden v razvojnem okolju prenaša med različnimi sistemi.
2. OSNOVE PROGRAMIRANJA	
• spozna zgradbo programa, • zna deklarirati spremenljivke in konstante, • razlikuje podatkovne tipe in zna izbrati najprimernejšega, • pozna aritmetične in logične operacije, • uporablja krmilne stavke, • problem strukturira na manjše probleme in manjše probleme rešuje z uporabo metod, • pozna metode z različnimi podpisi, • razlikuje med klicem parametrov metode po vrednosti in po referenci, • nauči se uporabljati eno in več dimenzionalna polja.	• napiše osnovne programe v izbranem programskem jeziku; v osnovnih programih poišče in odpravi sintaktične napake, • deklarira in uporablja enostavne podatkovne tipe; določi napake, ki se pojavijo zaradi predstavitve števil v računalniku in jih ustrezno obrazloži, • uporablja osnovne aritmetične in logične operacije, • v programu uporablja pogojne in brezpogojne vejitve, • ponavljajoče dele programa zapiše v zanki, • daljši program razdeli na smiselno povezane enote in dele kode zapiše v metode; metode ustrezno kliče v programu; napiše metode z različnim podpisi in jih smiselno uporabi, • v programu deklarira polje in ga ustrezno uporablja; našteje omejitve pri uporabi

	polj in pozna prednosti uporabe polj pred drugimi podatkovnimi strukturami.
3. OBJEKTNO ORIENTIRANO PROGRAMIRANJE	
• zna napisati razred kot predlogo objekta, • iz razreda ustvari izvode in jim priredi lastnosti in obnašanje, • razume razliko med razrednimi spremenljivkami in spremenljivkami izvoda, • kontrolira dostop do podatkov v razredu, • uporabi dedovanje, • spozna abstraktne razrede in vmesnike, • loči med virtualnimi in ne-virtualnimi metodami, • ustrezno uporablja polimorfizem.	• napiše preproste razrede, iz razredov izpelje izvode; lastnosti in obnašanje konkretnega objekta abstrahira v razred; uporablja konstruktorje razredov, zna napisati destruktor, • spremenljivke izvoda in razreda skrije v definiciji razreda, dostop do njih omogoča prek javnih metod, • načrtuje hierarhijo razredov in iz načrta napiše programsko kodo, • opiše razlike med abstraktnim razredom in vmesnikom; na različnih primerih pridobi izkušnje uporabe enega ali drugega, • razloži pomen poznega vezanja; v programu prepozna virtualne in nevirtualne metode; našteje primere uporabe polimorfizma.
4. OBRAVNAVA IZJEM	
• razlikuje med napakami v programu, ki se pojavijo med prevajanjem, med izvajanjem in med logičnimi napakami, • odpravlja napake v sintaksi, • pravilno obravnava izjeme, • išče logične napake z razhroščevalnikom.	• v programu odpravi sintaktične napake glede na sporočila prevajalnika, • testira svoj delujoč program, • testira poljuben program in izdela poročilo o izjemah in napakah, ki jih je v programu odkril, • napisanemu programu doda obravnavo izjem, • uporabi razrede izjem razvojnega okolja in napiše lastne razrede izjem.
5. GRAFIČNI VMESNIK	
• napiše grafični vmesnik brez uporabe razvojnega okolja, • načrtuje grafični vmesnik z uporabo razvojnega okolja, • ustrezno obravnava dogodke na posameznem kontrolniku, • zna izdelati svoje kontrolnike na podlagi že izdelanih.	• napiše program, v katerem loči logiko programa od uporabniškega vmesnika; za reševanje istega problema uporabi konzolni in grafični vmesnik, • grafični vmesnik izdela v razvojnem okolju; obravnava različne dogodke, • na podlagi izdelanega grafičnega vmesnika izdela lasten grafični kontrolnik in ga uporabi na primeru.
6. DATOTEKE	

• spozna različne vrste datotek, • zna uporabljati besedilne datoteke.	• uporablja knjižnice razredov, ki omogočajo upravljanje z datotekami in mapami, • napiše program, s katerim bere iz besedilne datoteke in piše v besedilno datoteko, • opiše direktne datoteke.
PROGRAMIRANJE II	
1. REKURZIJA	
• pozna probleme, ki jih rešujemo rekurzivno, • določi robni pogoj rekurzije in zna zapisati rešitev problema z uporabo rešitve manjšega problema.	• napiše iterativno in rekurzivno rešitev danega problema, • primerja rešitvi glede porabe časa za reševanje problema in glede porabe računalniškega časa in prostora, • za dani problem izbere med iterativno in rekurzivno različico rešitve, • oceni in pojasni ustreznost izbora.
2. RAZVRŠČANJE PODATKOV	
• pozna postopke razvrščanja podatkov, • primerja različne algoritme za razvrščanje podatkov, • pozna postopke iskanja elementa v polju.	• opiše postopek urejanja z izbiranjem, z vstavljanjem in s premenami, • napiše programe za vse tri vrste urejanja, • pozna algoritem quicksort; napiše program, ki uredi podatke z algoritmom quicksort, • napiše program, ki meri časovne zahtevnosti različnih algoritmov, • uporabi že izdelane metode za urejanje podatkov.
3. UPORABA KNJIŽNIC RAZREDOV	
• pozna osnovne metode baznega razreda vseh drugih razredov (Object), • spozna sklad in vrsto, • sklepa o različnih možnostih uporabe vmesnika za enumeracijo, • nauči se uporabljati razpršilne tabele (slovarje).	• v aplikaciji kliče metode baznega razreda in jih po potrebi ustrezno prepiše, • uporablja sklad in vrsto; nad njima izvaja osnovne operacije, • napiše program, v katerem definira svojo zbirko in jo oštevilči s poljubnim indeksom, • po navodilih izdela lasten slovar in poišče različne možnosti uporabe podatkovne strukture razpršilne tabele.
4. VMESNIK ZA DOSTOP DO ZBIRK PODATKOV	
• našteje razloge za izdelavo aplikacije za dostop do zbirke podatkov,	• opiše objektni model razredov, ki omogočajo dostop do zbirke podatkov,

• pozna postopke za dostop do zbirke podatkov, • oceni, katere rešitve je smiselno izvesti v zbirki podatkov, katere pa v uporabniški aplikaciji.	• napiše program, ki omogoča različne vpoglede v podatke v zbirki, • program dopolni z vstavljanjem, popravljanjem in brisanjem zapisov v zbirki prek uporabniškega vmesnika.
5. PROGRAMIRANJE ZA SPLET	
• razlikuje med okenskimi in spletnimi aplikacijami, • izdela aktivno spletno stran, • dostopa do zbirke podatkov prek spletne aplikacije, • spletno aplikacijo namesti na spletni strežnik.	• izdela preprosto aktivno spletno stran, • pri izdelavi spletne strani uporablja različne kontrolnike, • preverja vnos uporabnika spletne strani in ustrezno obvešča o napakah, • dogodke in odgovore strežniku obravnava, • uporablja razrede, ki omogočajo vzdrževanje stanja v spletni aplikaciji, • napiše spletno aplikacijo za dostop do zbirke podatkov; razloži razliko med dostopom do zbirke podatkov v okenski in spletni aplikaciji, • spletno aplikacijo iz razvojnega okolja namesti na spletni strežnik.

5. OBVEZNOSTI ŠTUDENTOV IN POSEBNOSTI V IZVEDBI

Predmet je razdeljen v dva sklopa. Prvi sklop (Programiranje I) se izvaja v 1. letniku, drugi sklop (Programiranje II) pa v 2. letniku.

1. letnik:
Število kontaktnih ur: 84 ur (36 ur predavanj, 48 ur vaj).
Število ur samostojnega dela: 126 ur (54 ur študij literature, 30 ur vaj, 42 ur seminarska naloga).
Skupaj 210 ur dela študenta (7 KT).
Obvezna je prisotnost na vajah, izdelava in predstavitev seminarske naloge ter pisni izpit.

2. letnik:
Število kontaktnih ur: 84 ur (36 ur predavanj, 48 ur vaj).
Število ur samostojnega dela: 96 ur (30 ur študij literature, 24 ur vaj, 42 ur seminarska naloga).
Skupaj 180 ur dela študenta (6 KT).
Obvezna je prisotnost na vajah, izdelava in predstavitev seminarske naloge ter pisni izpit.


Evropski
Socialni
Sklad