Státní závěrečné zkoušky	Akad. rok 2018/2019
Bakalářský studijní program:	Inženýrská informatika
Obor:	Softwarové inženýrství

Programovací techniky a návrh software

1. Datové struktury s dynamickým přidělováním paměti: dynamicky rostoucí pole, lineární seznam, hash tabulka, binární strom. Odvoďte vztahy pro prostorovou složitost. Porovnejte asymptotické časové složitosti operací vkládání (na začátek / na konec a uprostřed stávajících dat), vyhledávání a výmaz jednoho prvku. U jednotlivých datových struktur uveďte možné metody pro rychlé řazení všech prvků.
2. Řadicí algoritmy Quick Sort, Heap Sort a Merge Sort. Vysvětlete princip jejich funkce a zdůvodněte asymptotické časové složitosti.
3. Vyhledávání podřetězců v textu: vysvětlete algoritmus „hrubé síly“ (naivní), Karp-Rabinův a Quick Search (Sunday) algoritmus a porovnejte jejich časové složitosti.
4. Zapouzdření objektů v objektově orientovaném programování, gettery a settery, použití, výhody a nevýhody.
5. Dědičnost kódu, použití, výhody, nevýhody a srovnání se skládáním objektů (kompozicí).
6. Dědičnost rozhraní (interface), použití, výhody a nevýhody. Polymorfismus a překrývání virtuálních metod, abstraktní třída, abstraktní a virtuální metody.
7. Návrhový vzor Singleton a technika Dependency Injection. Architektonický vzor uživatelského rozhraní Model–view–viewmodel (MVVM), jeho použití, výhody a nevýhody.
8. Základní principy tvorby aplikací v jazyce C/C++, výhody a nevýhody jazyka C/C++ ve srovnání s jinými moderními programovacími jazyky (C#, Java, Python, JavaScript, případně dalšími). Proces překladu zdrojového kódu jazyka C/C++, preprocessing zdrojového kódu. Dělba zdrojového kódu na moduly.
9. Základní a rozšířené datové typy jazyka C/C++, proměnné, jejich přetypování a implicitní a explicitní konverze. Pole, struktury, uniony. Základní typy paměťových prostorů podporovaných jazykem C/C++, správa dynamické paměti.
10. Příkazy a klíčová slova jazyka C/C++. Smyčky a řízení toku programu. Funkce a procedury. Podpora pro I/O operace (konzolové, souborové, …) ve standardní knihovně jazyka.
11. Podpora OOP v programovacím jazyce C++. Podpora šablon a generických tříd v jazyce C++, určení a způsob práce s generickými datovými typy. Standardní knihovna STL, podpora pro generické datové kontejnery.
12. Vysvětlete způsob implementace aplikace – více-vláknového serveru, schopného obsloužit N současných klientů komunikujících protokolem TCP/IP. Jakými způsoby je možné implementovat obranu proti DoS/DDoS útokům typu „SYN Flood Attack“?
13. Popište protokol SOAP a porovnejte jeho výhody a nevýhody ve srovnání s jednoduchými HTTP web-API, které komunikují např. pomocí uživatelsky definovaných zpráv ve formátu JSON.
14. Vývoj mobilních aplikací – definice metod vývoje mobilních aplikací, základní charakteristika, výhody a nevýhody jednotlivých přístupů, výčet softwarových nástrojů a programovacích jazyků.
15. Vývoj mobilních aplikací pomocí HTML5 – charakteristika, architektura aplikace, vhodné vývojové frameworky, možnosti propojení funkcionality z webového kontextu na nativní úroveň, výhody a nevýhody oproti nativnímu přístupu.

16. Nativní vývoj pro Android – popis základních tříd pro implementaci mobilní aplikace, charakteristika souboru AndroidManifest.xml. Způsob oddělení statických zdrojů aplikace (Resources) od zdrojového kódu a jeho výhody. Definice životního cyklu aktivity a způsob spouštění aktivit či jiných komponent aplikace.
17. Jednočipové mikropočítače – struktura, paralelní porty, sériové porty, komunikace po sběrnici, čítače/časovače, přerušovací systém, mikropočítače s procesorem HCS08, ARM Cortex-M, programovací jazyky, assembler, softwarové a hardwarové prostředky pro interní diagnostiku správného běhu mikropočítače.
18. Princip činnosti operačních systémů pro práci v reálném čase, charakteristika systému RTOS, jeho softwarové součásti, jejich význam a použití, datové struktury, plánovací strategie, „tik“ OS. Procesy, datový vektor procesu, prostředky pro předávání informací mezi procesy a pro synchronizaci běhu procesů, datový vektor schránky.
19. Metody sběru a analýzy požadavků. Rozdělení požadavků. Význam modelů případů užití v UML – aktéři, případy užití, vazby mezi aktéry, vazby mezi případy užití. Principy tvorby scénáře případu užití, větvení scénářů pomocí klíčových slov, princip hlavního a alternativního scénáře. Vztah požadavků a případů užití.
20. Objektový přístup a diagram tříd v UML. Význam a definice třídy v UML, typy asociací mezi třídami, násobnost. Metody nalezení tříd, definice operací a atributů.
21. Sekvenční diagram v UML a jeho význam, význam pro realizaci případu užití. Synchronní a asynchronní zprávy, struktura zprávy. Princip self-message zprávy, příklad modelu.
22. Metody odhadování rozsahu a pracnosti softwarového projektu. Principy a postupy odhadu. Charakteristika vybraných metod – zaměření na Use Case Points. Ilustrace výpočetního postupu.
23. Základní řetězec omyl -> chyba -> selhání, cíle testování, základní testovací proces, 7 principů testování, testovací úrovně a rozdělení typů testů.
24. Rozdělení technik návrhu testů - testování pomocí černé, bílé skříňky a testování založené na zkušenostech, popis jednotlivých technik včetně praktické ukázky. Metody statického testování.
25. Role testování v různých životních vývojových cyklech (klasický sekvenční model vs. iterativně-inkrementální model, jednotlivé typy). Kategorizace nástrojů využívaných pro testování software. Výhody a nevýhody využití nástrojů, včetně popisu rizik spojených s využitím nástrojů při testování.