СОФИЙСКИ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ“
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА

КОНСПЕКТ

ЗА
ДЪРЖАВЕН ИЗПИТ ЗА ЗАВЪРШВАНЕ НА
ОБРАЗОВАТЕЛНО-КВАЛИФИКАЦИОННА
СТЕПЕН „БАКАЛАВЪР“

СПЕЦИАЛНОСТ

„КОМПЮТЪРНИ НАУКИ“

Промените в конспекта за ДИ са приети с
решения на ФС: протокол № 1/24.01.2022 г.
протокол № 2/28.02.2022 г.
протокол № 2/24.02.2025 г.
протокол № 6/30.06.2025 г.

СОФИЯ • 2025

СЪДЪРЖАНИЕ

УКАЗАНИЯ ЗА ПРОВЕЖДАНЕТО НА ДЪРЖАВЕН ИЗПИТ ЗА ЗАВЪРШВАНЕ НА ОБРАЗОВАТЕЛНО-КВАЛИФИКАЦИОННА СТЕПЕН „БАКАЛАВЪР”

СПЕЦИАЛНОСТ „КОМПЮТЪРНИ НАУКИ”

При явяване на държавен изпит всеки студент е длъжен да носи студентската си книжка, да се яви навреме пред предварително оповестената зала и да спази указанията на квесторите за настаняване в залата.

Държавният изпит по специалност *„Компютърни науки”* е в две части, които се провеждат в два дни. През първия ден изпитът е практически (решаване на задачи). На втория ден изпитът е теоретичен.

По време на всяка една част от изпита листата за писане са осигурени и подпечатани от ФМИ, други не се внасят. Пише се само с химикал - задължително син или черен цвят. Молив може да се използва само за чертежи.

По време на изпита може да се използва официално издадено копие на конспекта (получава се от квесторите; собствено копие не може да се внася). Всички други пособия са забранени.

Забранено е използването на електронни устройства от всякакъв вид. Необходимо е всички внесени мобилни устройства и компютърна техника да бъдат изключени преди започване на изпита и да бъдат оставени на определените за целта места. Намирането при студентите на нерегламентирани помощни средства се счита за опит за преписване. По време на изпита не се водят разговори, не се пуши и не се излиза от залата.

Работите се оценяват от комисия. Практическият и теоретичният изпит се оценяват поотделно. При положение, че и на двата изпита оценката е по-голяма или равна на 3.00, то крайната оценка от държавния изпит е закръглената по правилата средно аритметична оценка от двата изпита. В противен случай оценката е слаб (2.00). Оценката се закръгля до втори знак след десетичната запетая**.** Оценките са окончателни и не подлежат на преразглеждане.

Според правилника на СУ студентите нямат право на явяване за повишаване на оценка от ДИ, ако той е бил успешно положен. Напомняме на студентите, че според ЗВО за продължаване на образованието в ОКС „Магистър” **(срещу заплащане)** е необходима оценка най-малко „добър” от дипломата за ОКС „Бакалавър”.

КОНСПЕКТ ЗА ДЪРЖАВЕН ИЗПИТ ЗА СПЕЦИАЛНОСТ „КОМПЮТЪРНИ НАУКИ“

ОСНОВИ НА КОМПЮТЪРНИТЕ НАУКИ

1. Множества. Декартово произведение. Релации. Функции.
2. Основни комбинаторни принципи и конфигурации. Рекурентни уравнения.
3. Графи. Дървета. Обхождания на графи.
4. Характеризация на регулярните езици. Теорема на Майхил-Нероуд.
5. Лема за разрастването за контекстносвободни езици. Незатвореност на класа на контекстносвободните езици относно сечение и допълнение.
6. Сортиране чрез сравнения във време $O(n \lg n)$.
7. Минимални покриващи дървета.
8. Най-къси пътища в тегловни графи.

ЯДРО НА КОМПЮТЪРНИТЕ НАУКИ

9. Компютърни архитектури – Формати на данните. Вътрешна структура на централен процесор – блокове и конвейерна обработка, инструкции.
10. Структура и йерархия на паметта. Сегментна и странична преадресация. Система за прекъсване – приоритети и обслужване.
11. Файлова система. Функции, структура и реализация.
12. Управление на процеси и междупроцесни комуникации.
13. Компютърни мрежи и протоколи – OSI модел. Маршрутизация. Протоколи IPv4, IPv6, TCP, DNS.
14. Процедурно програмиране – основни конструкции.
15. Процедурно програмиране – указатели, масиви и рекурсия.
16. Обектно-ориентирано програмиране. Основни принципи. Класове и обекти. Наследяване и капсулация.
17. Обектно-ориентирано програмиране. Подтипов и параметричен полиморфизъм. Множествено наследяване.
18. Структури от данни. Стек, опашка, списък, дърво. Основни операции върху тях. Реализация
19. Функционално програмиране. Обща характеристика на функционалния стил на програмиране. Дефиниране и използване на функции. Модели на оценяване. Функции от по-висок ред.
20. Функционално програмиране. Списъци. Потоци и отложено оценяване.
21. Синтаксис и семантика на предикатното смятане от първи ред.
22. Изводимост и компютърно генериране на доказателства
23. Бази от данни. Релационен модел на данните.
24. Бази от данни. Нормални форми.

25. Изкуствен интелект: Пространство на състоянията – определение, характеристики на състоянията, търсене в пространство на състояния

26. Съвременни софтуерни технологии.

27. Архитектури на софтуерни системи.

МАТЕМАТИКА И ПРИЛОЖЕНИЯ

28. Уравнения на права в равнината.

29. Уравнения на права и равнина в пространството.

30. Симетрични оператори в крайномерни евклидови пространства. Основни свойства. Теорема за диагонализация.

31. Симетрична и алтернативна група. Теорема на Кейли. Теорема за хомоморфизмите на групи.

32. Теорема на Ферма. Теореми за средните стойности (Рол, Лагранж и Коши). Формула на Тейлър.

33. Определен интеграл. Дефиниция и свойства. Интегруемост на непрекъснатите функции. Теорема на Нютон - Лайбниц.

34. Итерационни методи за решаване на нелинейни уравнения.

35. Случайни величини с дискретни разпределения – биномно, геометрично и Поасоново разпределение. Задачи, в които възникват.

АНОТАЦИИ НА ВЪПРОСИТЕ

1. Множества. Декартово произведение. Релации. Функции.

Аксиоматизация на множествата – аксиоми за обема, отделянето, степенното множество и индуктивно генерираните множества. Математическа индукция. Основни операции върху множества и техните свойства. Наредена двойка и наредена n-орка. Декартово произведение и обобщено Декартово произведение на множества. Релация над n домейна. Свойства на бинарните релации. Релации на еквивалентост и класове на еквивалентност. Релации на частична наредба. Диаграми на Хасе. Релации на пълна наредба. Минимален и максимален елемент в релация на частична наредба. Влагане на частична наредба в пълна наредба – топологично сортиране. Частични и тотални функции. Инекции, биекции и сюрекции. Дефиниция на крайно множество и на кардиналност на крайно множество. Дефиниция на изброимо безкрайно множество. Принцип на Дирихле.

Примерни задачи

1. Да се докаже, че множествата, описани от дадени изрази, са равни: било чрез таблица, било чрез използване на свойствата на операции върху множества.
2. Да се докаже или опровергае, че дадена бинарна релация има дадено свойство.
3. Да се определят класовете на еквивалентност на дадена релация на еквивалентност.
4. Да се докаже по индукция дадено твърдение върху естествените числа.
5. Да се докаже твърдение чрез принципа на Дирихле.

Литература: [12], [33], [40].

2. Основни комбинаторни принципи и конфигурации. Рекурентни уравнения.

Формулировки на принципите на изброителната комбинаторика – принцип на Дирихле, принцип на биекцията, принцип на събирането, принцип на изваждането, принцип на умножението, принцип на делението, принцип за включване и изключване (с доказателство). Основните комбинаторни конфигурации: с или без наредба, с или без повтаряне. Извеждане на формулите за броя на основните комбинаторни конфигурации. Биномни коефициенти и теорема на Нютон. Доказателства на комбинаторни тъждества чрез комбинаторни разсъждения (принцип на двукратното броене). Алгоритъм за решаване на линейни рекурентни уравнения с константни коефициенти – хомогенни и нехомогенни.

Примерни задачи

1. Да се намери броят на частичните функции, на тоталните функциите, на инекции и на сюрекциите от крайно множество A в крайно множество B.
2. Да се намери броят на различните целочислени решения на уравнение от вида
3. $x_1 + x_2 + \ ... \ + x_n = M$, така че всички x_i са неотрицателни и трябва да удовлетворяват различни други условия, например $x_i \geq c_i$.
4. Да се реши зададено линейно (хомогенно или нехомогенно) рекурентно уравнение с константни коефициенти.

Литература: [12], [33], [40].

3. Графи. Дървета. Обхождания на графи.

Дефиниции за краен ориентиран (мулти)граф и краен неориентиран (мулти)граф. Дефиниции за път (цикъл) в ориентиран и неориентиран мултиграф. Свързаност и свързани компоненти на граф. Дефиниция на дърво и кореново дърво. Доказателство, че всяко кореново дърво е дърво и $|V| = |E| + 1$. Покриващо дърво на граф. Обхождане на граф в ширина и дълбочина. Ойлерови обхождания на мултиграф. Теореми за съществуване на Ойлеров цикъл (с доказателство) и Ойлеров път.

Примерни задачи

1. Да се построи покриващо дърво на зададен граф – в ширина или дълбочина.
2. Да се построи Ойлеров цикъл (или път) в зададен мултиграф или да се докаже, че такъв цикъл (или път) не съществува.
3. Да се разбие множеството на ребрата на неориентиран граф на минимален брой пътища, никои два от които нямат общо ребро.

Литература: [12], [33], [40].

4. Характеризация на регулярните езици. Теорема на Майхил-Нероуд

1) Дефинират се понятията регулярен език, детерминиран краен автомат и автоматен език. Формулира се теоремата на Клини.
2) За език L се дефинира релацията $\approx_L$ по един от следните два начина:
   - въвежда се $\alpha^{-1}(L)$ и се полага $\alpha \approx_L \beta \Leftrightarrow \alpha^{-1}(L) = \beta^{-1}(L)$,
   - или като релацията на Майхил-Нероуд за езика L.
3) Доказва се, че $\approx_L$ е релация на еквивалентност и че е дясно инвариантна.
4) Доказва се, че ако релацията $\approx_L$ има индекс n, то L се разпознава от тотален краен детерминиран автомат с n състояния.
5) Доказва се, че ако L се разпознава от тотален краен детерминиран автомат с n състояния, то индексът на релацията $\approx_L$ е не повече от n.
6) Доказва се, че един език L е регулярен точно тогава, когато релацията $\approx_L$ има краен индекс.

Примерни задачи

- По даден регулярен език да се построи минимален тотален краен детерминиран автомат. Решението трябва да се обоснове.
- Да се определи, с обосновка, дали даден език е регулярен.

Литература: [34], [38], [43].

5. Лема за разрастването за контекстносвободни езици. Незатвореност на класа на контекстносвободните езици относно сечение и допълнение.

1) Дефинират се понятията контекстносвободна (безконтекстна) граматика, извод в контекстносвободна граматика, контекстносвободен език и дърво на извод в контекстносвободна граматика.
2) Затвореност на контекстносвободните езици относно регулярните операции (дават се конструкции, без доказателства).
3) Формулира се и се доказва лемата за разрастването за контекстносвободни езици.
4) Дава се пример с доказателство за неконтекстносвободен език.
5) Доказва се, че контекстносвободните езици не са затворени относно сечение и допълнение.

Примерни задачи

- По даден контекстносвободен език да се построи контекстносвободна граматика. Решението трябва да се обоснове.
- Да се определи, с обосновка, дали даден език е контекстносвободен

Литература: [34], [38], [43].

6. Сортиране чрез сравнения във време O(n lg n).

Двоична пирамида (binary heap): определение и свойства. Наивно построяване на двоична пирамида: доказателство за коректност и изследване на сложността по време. Бързо построяване на двоична пирамида, използващо функция Heapify: доказателство за

коректност и изследване на сложността по време. Сортиращ алгоритъм HEAPSORT: псевдокод, доказателство за коректност и изследване на сложността по време.
Сортиращ алгоритъм MERGESORT като пример за алгоритъм по схемата Разделяй и Владей. Псевдокод и доказателство за коректност на MERGESORT. Изследване на сложността по време на MERGESORT. Приложение на MERGESORT за намиране на броя на инверсиите в масив от числа, с доказателство за коректност.

Забележка: На изпита ще се падне или HEAPSORT, или MERGESORT.
Литература: [31].

7. Минимални покриващи дървета.
Дефиниция на задачата – какъв вид графи се разглеждат и какво се иска. Теоремата за съгласуваното множество (условия за нарастване на подмножество на МПД) с доказателство.

- Алгоритъм на Прим: псевдокод и доказателство за коректност. Изследване на сложността по време на алгоритъма на Прим при използване на различни структури от данни.
- Алгоритъм на Крускал: псевдокод, доказателство за коректност и изследване на сложността по време. Union-Find структури от данни и приложението им за ефикасната реализация на алгоритъма на Крускал.

Забележка: На изпита ще се падне или алгоритъмът на Прим, или алгоритъмът на Крускал.
Литература: [31].

8. Най-къси пътища в тегловни графи.
Задачата за намиране на най-къси пътища в тегловни графи – определение и варианти. Потенциални проблеми при наличието на отрицателни тегла.

- Алгоритъм на Дийкстра: вариант на задачата, псевдокод и доказателство за коректност. Изследване на сложността по време на алгоритъма на Дийкстра при използване на различни структури от данни.
- Алгоритъм за намиране на най-къси пътища в тегловен ориентиран ацикличен граф: вариант на задачата, псевдокод, доказателство за коректност и изследване на сложността по време. Относителни предимства пред алгоритъма на Дийкстра
- Алгоритъм на Белман–Форд: вариант на задачата, псевдокод, доказателство за коректност и изследване на сложността по време. Установяване на наличието на цикли с отрицателно сумарно тегло.
- Алгоритъм на Флойд–Уоршал: вариант на задачата, псевдокод, доказателство за коректност, изследване на сложността по време и сложността по памет.

Забележка: На изпита ще се падне или алгоритъмът на Дийкстра и алгоритъмът за намиране на най-къси пътища в тегловни ориентирани ациклични графи, или алгоритъмът на Белман–Форд и алгоритъмът на Флойд–Уоршал.
Литература: [31].

9. Компютърни архитектури – Формати на данните. Вътрешна структура на централен процесор – блокове и конвейерна обработка, инструкции.
Обща структура на компютрите и концептуално изпълнение на инструкциите, запомнена програма. Формати на данните – цели двоични числа, двоично-десетични числа, двоични числа с плаваща запетая, знакови данни и кодови таблици. Централен процесор –

регистри, АЛУ, регистри на състоянията и флаговете, блокове за управление, връзка с паметта, дешифрация на инструкциите, преходи.
Литература: [5], [36], [45].

10. Структура и йерархия на паметта. Сегментна и странична преадресация. Система за прекъсване – приоритети и обслужване.
Структура на основната памет. Йерархия – кеш, основна и виртуална памет. Сегметна и странична преадресация – селектор, дескриптор, таблици и регистри при сегметна преадресация; каталог на страниците, описател, стратегии на подмяна на страниците при странична преадресация. Система за прекъсване – видове прекъсвания, структура и обработка, конкурентност и приоритети, контролери на прекъсванията.
Литература: [5], [45], [50].

11. Файлова система. Функции, структура и реализация.
Анотация (примерна операционна система Linux):
Файлове и техните основни атрибути. Файловата система като абстракция за организация на информационните обекти:
– единно пространство от именувани обекти – структура, физическо пред- ставяне, точки на монтиране
– видове обекти – файлове, директории, устройства, информационни ка- нали
– основни функции – достъп до информационните обекти, изолация и за- щита на потребителите, управление на правата на достъп
Реализация на конкретна файлова система:
– стандарти за представяне – кратък обзор на ext2/ext3
– ускоряване на достъпа - кеширане на вход/изхода, отлагане на записа, алгоритъм на асансьора
– надеждна реализация - журнална файлова система, блокови устройства с поддържане на излишък: RAID1, RAID5
Функции за работа с файлове в стандарта POSIX: open() close() read() write() lseek()
Литература: [42, четвърта глава], [48], [49, глави 10-12].

Примерни задачи
Задача 1. Двоичните файлове f1 и f2 съдържат 32 битови числа без знак (uint32_t). Файлът f1 съдържа n двойки числа, нека i-тата двойка е $\langle x_i, y_i \rangle$. Напишете програма, която извлича интервалите с начало x_i и дължина y_i от файла f2 и ги записва залепени в изходен файл f3.

Пример: f1 съдържа 4 числа (2 двойки): 30000 20 19000 10
Програмата записва в f3 две поредици 32-битови числа, взети от f2 както следва:
Най-напред се записват числата, които са на позиции 30000, 30001, ... 30019.
След тях се записват числата от позиции 19000, 19001, ... 19009.
Забележка: С пълен брой точки ще се оценяват решения, които работят със скорост, пропорционална на размера на изходния файл f3.

Задача 2. Напишете програма на C, която приема два параметъра, имена на входен и изходен файл. Примерно изпълнение:
```
./main f1.bin f2.bin
```

Файловете са двоични (binary) файлове, съдържащи 8-битови числа без знак (`uint8_t`) и максималният допустим размер на входния файл f1.bin е до 2 GB. Програмата трябва да сортира във възходящ ред файла f1.bin и да записва резултата във f2.bin.

Примерен f1.bin:
```
0000000: 2550 4446 2d31 2e34 0a25 c7ec 8fa2 0a35
0000010: 2030 206f 626a 0a3c 3c2f 4c65 6e
```

Примерен f2.bin:
```
0000000: 0a0a 0a20 2025 252d 2e2f 3031 3435 3c3c
0000010: 4446 4c50 6265 6a6e 6f8f a2c7 ec
```

Забележка: С пълен брой точки ще се оценяват решения, които работят със скорост, пропорционална на размера на входния файл f1.

Задача 3. Напишете програма на C, която приема три параметъра, имена на двоични файлове. Примерно изпълнение:
```
./main patch.bin f1.bin f2.bin
```

Файловете f1.bin и f2.bin се третират като двойни файлове, състоящи се от байтове (`uint8_t`). Файлът `patch.bin` е двоичен файл, състоящ се от наредени тройки от следните елементи (и техните типове):
• *отместване* `uint16_t`
• *оригиналенбайт* `uint8_t`
• *новбайт* `uint8_t`

Програмата да създава файла `f2.bin` като копие на файла `f1.bin`, но с отразени промени на базата на файла `patch.bin`, при следният алгоритъм:

- за всяка наредена тройка от `patch.bin`, ако на съответното *отместване* (в байтове), спрямо началото на файла е записан байта *оригиналенбайт*, в изходния файл се записва *новбайт*. Ако не е записан такъв *оригиналенбайт* или такова отместване не съществува, програмата да прекратява изпълнението си по подходящ начин
- всички останали байтове се копират директно

Забележка: Наредените тройки във файла `patch.bin` да се обработват последователно.

Примерен `f1.bin`:
```
00000000: f5c4 b159 cc80 e2ef c1c7 c99a 2fb0 0d8c  ...Y......../...
00000010: 3c83 6fed 6b46 09d2 90df cf1e 9a3c 1f05  <.o.kF.......<..
00000020: 05f9 4c29 fd58 a5f1 cb7b c9d0 b234 2398  ..L).X...{...4#.
00000030: 35af 6be6 5a71 b23a 0e8d 08de def2 214c  5.k.Zq.:......!L
```

Примерен `patch.bin`:
```
00000000: 0200 b159 3000 35af ...Y0.5.
```

Примерен `f2.bin`:
```
00000000: f5c4 5959 cc80 e2ef c1c7 c99a 2fb0 0d8c  ..YY......../...
00000010: 3c83 6fed 6b46 09d2 90df cf1e 9a3c 1f05  <.o.kF.......<..
00000020: 05f9 4c29 fd58 a5f1 cb7b c9d0 b234 2398  ..L).X...{...4#.
00000030: afaf 6be6 5a71 b23a 0e8d 08de def2 214c  ..k.Zq.:......!L
```

12. Управление на процеси и междупроцесни комуникации.
Основни системни примитиви за управление на процеси. Създаване на процес. Изпълнение на програма. Завършване на процес. Синхронизация със завършването на процеса-син. Права на процеси – потребителски идентификатори на процес. Групи

процеси и сесия. Механизми за междупроцесни комуникации. Сигнали. Програмни канали. IPC пакет на UNIX System V: Обща памет. Семафори. Съобщения.

Примерни задачи: задачи, съответни на съдържанието на въпроса.
Литература: [21], [46].

13. Компютърни мрежи и протоколи – OSI модел. Маршрутизация. Протоколи IPv4, IPv6, TCP, DNS.

OSI модел – най-обща характеристика на нивата, съпоставяне с модела TCP/IP. Разпределена маршрутизация – алгоритми с дистантен вектор и следене на състоянието на линията. IPv4 адресация – класова и безкласова. Основни характеристики на протокол IPv6. TCP – процедура на трикратно договаряне. Основни характеристики на протоколи DNS (резолвинг на имената по IPv4 и IPv6).
Литература:[3], [35], [51].

14. Процедурно програмиране – основни конструкции.

1. Принципи на структурното програмиране.
2. Управление на изчислителния процес. Основни управляващи конструкции – условни оператори, оператори за цикъл.
3. Променливи – видове: локални променливи, глобални променливи; инициализация на променлива; оператор за присвояване.
4. Функции и процедури. Параметри – видове параметри. Предаване на параметри – по име и по стойност. Типове и проверка за съответствие на тип

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [24], [27], [47].

15. Процедурно програмиране – указатели, масиви и рекурсия.

Изложението по въпроса трябва да включва следните по-съществени елементи:

1. Указатели и указателна аритметика.
2. Едномерни и многомерни масиви. Основни операции с масиви – индексиране. Сортиране и търсене в едномерен масив – основни алгоритми.
3. Символни низове. Представяне в паметта. Основни операции със символни.
4. Рекурсия – пряка и косвена рекурсия, линейна и разклонена рекурсия.

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [24], [27], [47].

16. Обектно-ориентирано програмиране. Основни принципи. Класове и обекти. Наследяване и капсулация.

1. Абстракция със структури от данни. Класове и обекти. Декларация на клас и декларация на обект. Основни видове конструктори. Управление на динамичната памет и ресурсите (“RAII”). Методи – декларация, предаване на параметри, връщане на резултат.
2. Наследяване. Производни и вложени класове. Достъп до наследените компоненти. Капсулация и скриване на информацията. Статични полета и методи.

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [23], [24], [26], [47].

17. Обектно-ориентирано програмиране. Подтипов и параметричен полиморфизъм. Множествено наследяване.

1. Виртуални функции и подтипов полиморфизъм. Динамично свързване. Абстрактни методи и класове. Масиви от обекти и от указатели към обекти.
2. Параметричен полиморфизъм. Шаблони на функция и на клас.
3. Множествено наследяване.

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [23], [24], [26], [47].

18. Структури от данни. Стек, опашка, списък, дърво. Основни операции върху тях. Реализация

1. Структури от данни – дефиниране на понятието.
2. Списък. Логическо описание. Списък с една и две връзки. Характеристики на реализациите с една и две връзки. Сложност на операциите по добавяне, премахване и намиране на елемент. Дефиниране на клас за списък, използващ една от реализациите.
3. Стек. Логическо описание. Характеристики на статичната, динамичната и свързаната реализация. Сложност на операциите по добавяне и премахване на елемент. Дефиниране на клас за стек, използващ една от реализациите.
4. Опашка. Логическо описание. Характеристики на статичната, динамичната и свързаната реализация. Сложност на операциите по добавяне и премахване на елемент. Дефиниране на клас за опашка, използващ една от реализациите.
5. Дървовидни структури от данни – кореново дърво и двоично кореново дърво. Логическо описание. Начини за представяне в паметта. Дефиниране на клас, реализиращ кореново дърво или двоично кореново дърво.
6. Двоично кореново дърво за търсене. Логическо описание. Начини за представяне в паметта. Сложност на операциите по добавяне, премахване и търсене на елемент. Дефиниране на клас реализиращ двоично кореново дърво за търсене.

Забележка: За изпита ще бъдат избрани две от посочените структури, които да бъдат описани.
Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [11], [15], [25], [28].

19. Функционално програмиране. Обща характеристика на функционалния стил на програмиране. Дефиниране и използване на функции. Модели на оценяване. Функции от по-висок ред.

1. Характерни особености на функционалния стил на програмиране. Основни компоненти на функционалните програми. Примитивни изрази. Средства за комбиниране и абстракция. Оценяване на израз. Дефиниране на функция и оценяване на приложение на функция. Модели на оценяване. Апликативно (стриктно, call-by-value) и нормално (лениво, call-by-name) оценяване.
2. Функции от по-висок ред. Функциите като параметри и оценки на обръщения към функции. Анонимни (ламбда) функции.

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [14], [22], [29], [53].

20. Функционално програмиране. Списъци. Потоци и отложено оценяване.

1. Списъци. Представяне. Основни операции със списъци. Функции от по-висок ред за работа със списъци.
2. Безкрайни потоци и безкрайни списъци. Основни операции и функции от по-висок ред. Отложено оценяване. Работа с безкрайни потоци.

Забележка: По този въпрос е възможно да бъдат дадени задачи за практическата част на изпита.
Литература: [14], [22], [29], [53].

21. Синтаксис и семантика на предикатното смятане от първи ред.

Дефинират се понятията: език на предикатното смятане от първи ред, терм, формула, област на действие на квантор, свободна и свързана променлива за формула, затворена формула.
Дефинират се понятията: структура за предикатен език от първи ред, оценка на индивидните променливи, вярност на формула в структура при оценка, тъждествена вярност на формула в структура, предикатна тавтология.
Доказва се, че верността на формула в структура при оценка не зависи от стойността на променливите, които не се срещат като свободни във формулата.
Формулира се твърдение за стойността при замяна на променливи с термове (лема за субституциите) и се доказва в частния случай на безквантορна формула.

Примерни задачи: определимост на свойства в дадена структура, показване на изпълнимост на множество от предикатни формули чрез посочване на модел.
Литература: [13], [19], [37].

22. Изводимост и компютърно генериране на доказателства

Дефинират се понятията: литерал, дизюнкт, верен в структура дизюнкт, изпълнимо множество от дизюнкти.
Дефинират се понятията: (предикатна либерална) резолвента, резолютивен извод.
Формулират се теоремите за коректност и за пълнота на резолютивната изводимост. Доказва се теоремата за коректност.

Примерни задачи: дефиниране на предикати на Пролог, доказване на неизпълнимост на множество от предикатни формули с метода на резолюциите.
Литература: [13], [19], [37].

23. Бази от данни. Релационен модел на данните.

Релационен модел на данните: домейн; релация; кортежи; атрибути; схема на релация; схема на релационна база от данни; видове операции върху релационната база от данни; заявки към релационната база от данни.
Релационна алгебра: основни (обединение; разлика; декартово произведение; проекция; селекция) и допълнителни (сечение; съединение; естествено съединение) операции.

Примерни задачи: Съставяне на SQL заявки, DDL и DML команди, тригери.
Литература: [32].

24. Бази от данни. Нормални форми.

Нормални форми. Проектиране схемите на релационните бази от данни. Аномалии, ограничения, ключове. Функционални зависимости, аксиоми на Армстронг. Първа, втора, трета нормална форма, нормална форма на Бойс-Код. Многозначни зависимости; аксиоми

на функционалните и многозначните зависимости; съединение без загуба; четвърта нормална форма.

Примерни задачи: Привеждане на схема на базата от данни (при зададени функционални зависимости) към зададена нормална форма.
Литература [32].

25. Изкуствен интелект: Пространство на състоянията – определение, характеристики на състоянията, търсене в пространство на състояния
Локално търсещи алгоритми:

- Локално търсене в лъч;
- Изкачване на хълмове;
- Симулирано втвърдяване;
- Генетични алгоритми.

Задачи за удовлетворяване на ограничения: определение, търсене с възврат, намаляване на конфликтите.
Избор на следващ ход при игра за двама играчи: мини-макс процедура, алфа-бета отсичане.
Литература: [41].

26. Съвременни софтуерни технологии.
Софтуерен продукт и процес. Основни фази на софтуерните процеси. Модел на софтуерен процес – модел на водопада, модел на бързата разработка, еволюционни модели, прототипен модел, спираловиден модел. Сравнителна характеристика на моделите на софтуерни процеси. Гъвкави методи - Extreme Programming (XP) и SCRUM. Изисквания към софтуерните системи. Функционални и нефункционални изисквания. Анализ и проектиране на софтуерните изисквания. Езици за описание. UML. Верификация и валидация на софтуера. Управление на качеството.
Литература [9], [39], [44].

27. Архитектури на софтуерни системи.

a. Понятие за софтуерна архитектура. Структури и изгледи (structures and views) на архитектурата. Необходимост от софтуерни архитектури. Влияние на архитектурата върху проекта и организацията.
b. Изисквания към качеството (нефункционални изисквания) на системата.
c. Архитектурни стилове. Многослоен стил, Модел-изглед контролер (MVC), Поточни (Pipe-and-filter) архитектури. Архитектура ориентирана към услуги.
d. Проектиране на софтуерната архитектура. Процес за проектиране. Избор на подходящи структури. Последователност на създаване на архитектурата. Тактики (архитектурни решения) за постигане на желаните качествени показатели.
e. Документиране на софтуерната архитектура. Предназначение и основни елементи в документацията на архитектурата.

Литература: [30], [52].

28. Уравнения на права в равнината.
Векторно-параметрично уравнение на права. Координатни (скаларни) параметрични уравнения на права в равнината. Теорема за общо уравнение на права в равнината. Взаимни положения между две прави в равнината. Декартово уравнение на права в равнината. Нормално уравнение на права в равнината. Разстояние от точка до права. Полуравнини.
Литература: [20].

29. Уравнения на права и равнина в пространството.
Векторно-параметрично уравнение на равнина. Координатни (скаларни) параметрични уравнения на равнина в пространството. Теорема за общо уравнение на равнина. Взаимни положения между две равнини. Нормално уравнение на равнина. Разстояние от точка до равнина. Полупространства. Уравнения на права в пространството.
Литература: [20].

30. Симетрични оператори в крайномерни евклидови пространства. Основни свойства. Теорема за диагонализация.
Определение за симетричен оператор. Матрица на симетричен оператор спрямо ортонормиран базис. Всички характеристични корени на симетричен оператор са реални числа. Всеки два собствени вектора, съответстващи на различни собствени стойности, са ортогонални помежду си. Съществува ортонормиран базис на пространството, в който матрицата на симетричен оператор е диагонална.
Литература: [18].

31. Симетрична и алтернативна група. Теорема на Кейли. Теорема за хомоморфизмите на групи.
Симетрична група S_n – представяне на елементите като произведение на независими цикли. Спрягане на елементите на S_n. Транспозиции и представяне на елементите като произведение на транспозиции. Алтернативна група. Теорема на Кейли – всяка крайна група е изоморфна на подгрупа на симетричната група. Хомоморфизъм при групи, ядро и образ. Теорема за хомоморфизмите при групи.
Литература: [17].

32. Теорема на Ферма. Теореми за средните стойности (Рол, Лагранж и Коши). Формула на Тейлър.
Да се дефинира понятието локален екстремум на функция на една променлива. Да се формулира и докаже необходимо условие за локален екстремум за диференцируеми функции (теорема на Ферма).
Да се докажат следните теореми, формулирани общо за по-кратко.
Нека функцията f е непрекъсната в затворения интервал $[a,b]$ и притежава производна поне в отворения интервал (a,b). Да се докаже, че:
а) ако $f(a)=f(b)$, то съществува $c\in(a,b)$ такова, че $f'(c)=0$ (Рол);
б) съществува $c\in(a,b)$ такова, че $f(b)-f(a)=f'(c)(b-a)$ (Лагранж);
в) ако функциятаg е непрекъсната в затворения интервал $[a,b]$ и притежава производна поне в отворения интервал (a,b), като $g'(x)\neq 0,\ x\in(a,b)$, то съществува $c\in(a,b)$ такова, че

$$\frac{f(b)-f(a)}{g(b)-g(a)}=\frac{f'(c)}{g'(c)}$$ (Коши).

По отношение на твърдението във в) да се докаже, че при направените в него предположения имаме $g(a)\neq g(b)$.
За установяването на теоремата на Рол може да се използва без доказателство теоремата на Вайерщрас, според която всяка непрекъсната функция върху краен затворен интервал достига своите максимум и минимум.
Да се изведе формулата на Тейлър с остатъчен член във формата на Лагранж.
Литература: [6], [8], [10].

33. Определен интеграл. Дефиниция и свойства. Интегруемост на непрекъснатите функции. Теорема на Нютон - Лайбниц.

Да се дефинират последователно: разбиване на интервал, големи и малки суми на Дарбу. Да се установи, че при добавяне на нови точки в разбиването на интервала, големите суми на Дарбу не нарастват, а малките не намаляват (желателно е да се направи чертеж). Да се дефинира риманов интеграл чрез подхода на Дарбу.

Да се докаже, че дадена функция е интегруема по Риман тогава и само тогава, когато за всяко $\varepsilon > 0$ съществуват голяма сума на Дарбу S и малка сума на Дарбу такива, че $S - s < \varepsilon$. Като се използва тази теорема и теоремата на Кантор (без доказателство), според която всяка непрекъсната функция в краен и затворен интервал е равномерно непрекъсната, да се докаже, че всяка непрекъсната функция в краен и затворен интервал е интегруема по Риман.

Да се изброят (без доказателство) основните свойства на римановия интеграл. Да се докаже, че ако f е непрекъсната в $[a, b]$, то съществува $c \in [a, b]$ такова, че

$$\int_a^b f(x)dx = f(c)(b - a).$$

За установяването на това твърдение да се приложат (без доказателство) свойството за интегриране на неравенства и теоремата, че всяка непрекъсната функция в $[a, b]$ приема всички стойности между максимума и минимума си.

Да се докаже теоремата на Нютон-Лайбниц, т.е. ако f е непрекъсната в $[a, b]$, то за всяко $x \in [a, b]$ е изпълнено

$$\frac{d}{dx}\int_a^x f(t)dt = f(x).$$

Да се покаже как теоремата се използва за изчисляване на определени интеграли.

Литература: [6], [8], [10].

34. Итерационни методи за решаване на нелинейни уравнения.

Да се дефинира понятието *неподвижна точка* на изображението φ и да се докаже, че ако φ е непрекъснато изображение на интервала $[a, b]$ в себе си, то φ има поне една неподвижна точка в $[a, b]$. Да се покаже, че решаването на уравнението $f(x) = 0$ може да се сведе към намиране на неподвижна точка.

Да се дефинира понятието *свиващо изображение* и да се докаже, че ако φ е непрекъснато изображение на интервала $[a,\ b]$ в себе си и е свиващо с константа на Липшиц $q < 1$, то:

а) уравнението $x = \varphi(x)$ има единствен корен ξ в $[a, b]$;

б) редицата $\{x_n\}$ от последователни приближения (при произволно $x_0 \in [a, b]$ и $x_{n+1} = \varphi(xn)$, $n = 0, 1, 2, ...$, клони към ξ при $n \to \infty$, като $|x_n - \xi| \leq (b - a)q^n$, за всяко n.

Да се получи като следствие, че ако ξ е корен на уравнението $x = \varphi(x)$ и φ има непрекъсната производна в околност U на ξ, за която $|\varphi'(\xi)| < 1$, то при достатъчно добро начално приближение x_0 итерационният процес, породен от φ, е сходящ със скоростта на геометрична прогресия. Да се дефинира понятието *ред на сходимост*.

Да се дадат геометрична илюстрация, формула за последователните приближения и ред на сходимост при: метод на хордите, метод на секущите и метод на Нютон. Да се докаже, че при метода на хордите сходимостта е със скоростта на геометричната прогресия (при условие, че коренът е отделен в достатъчно малък интервал).

Литература: [1], [2], [16].

35. Случайни величини с дискретни разпределения – биномно, геометрично и Поасоново разпределение. Задачи, в които възникват.

Дефиниция на дискретно разпределение на случайна величина. Свойства на вероятностите (неотрицателност и нормираност). Дефиниция и свойства (без доказателства) на

пораждаща функция. За всяко от дадените разпределения: да се посочи пример, при който то възниква, да се изведе пораждащата функция и да се пресметнат математическото очакване и дисперсията му (със или без пораждаща функция).
Литература: [7, глави 2.3 (стр. 54-56), 3.2 (стр. 71-74), 6.1 (примери 1-4)], [4, тема: Дискретни разпределения.].

ЛИТЕРАТУРА

1. Андреев, А. и др., *Сборник от задачи по числени методи*, Университетско издателство „Св. Климент Охридски“, София, 1994.
2. Боянов, Б., *Лекции по числени методи*, Дарба, София, 1998.
3. Боянов, Л., К. Боянов и др., *Компютърни мрежи и телекомуникации*, изд. „Авангард Прима“, София, 2014.
4. Вындев, Д., *Записки по теория на вероятностите*, Електронно издание: https://intranet.fmi.uni-sofia.bg/index.php/s/ji3f3bMSFFZwc5f .
5. Горслайн, Дж., *Фамилия Intel 8086/8088*, Техника, София, 1990.
6. Джаков, П., Р. Леви, Р. Малеев, С. Троянски, *Диференциално и интегрално смятане*, ФМИ-СУ, София, 2004.
7. Димитров, Б., К. Янев, *Вероятности и статистика*, Университетско издателство „Св. Климент Охридски”, София, 1998.
8. Дойчинов, Д., *Математически анализ*, Университетско издателство „Св. Климент Охридски“, София, 1994.
9. Ескенази А., Манева Н., *Софтуерни технологии*, 2006.
10. Любенова, Е., П. Недевски, К. Николов, Л. Николова, В. Попов, *Ръководство по Математически анализ*, София, 1998.
11. Майерс, С., *По-ефективен C++. 35 нови начина да подобрите своите програми и проекти*, ЗеСТ Прес.
12. Манев, К., *Увод в дискретната математика*. Издателство „КЛМН – Красимир Манев“, София, пето редактирано издание, 2012, ISBN 954-535-136-5.
13. Метакидес, Д., А. Нероуд, *Принципи на логиката и логическото програмиране*, Виртех, София, 2000.
14. Нишева, М., П. Павлов, *Функционално програмиране на езика Scheme*, София, 2004.
15. Седжуик, Р., Алгоритми на C, ч.1-4: *Основи, структури от данни, сортиране, търсене*, СофтПрес.
16. Сендов, Бл., В. Попов, *Числени методи, I ч.,* Университетско издателство „Св. Климент Охридски“, София, 1996.
17. Сидеров, Пл., Чакърян, К., *Записки по алгебра: групи, пръстени, полиноми*, Веди, София, 2014.
18. Сидеров, Пл., Чакърян, К., *Записки по алгебра: линейна алгебра*, Веди, София, 2014.
19. Скордев, Д., *Логическо програмиране (записки)*. Електронно издание: http://www.fmi.uni-sofia.bg/fmi/logic/skordev/ln/lp/new/sydyrzha.htm.
20. Станилов, Гр., *Аналитична геометрия*, Софтех, София, 1998.
21. Стивенс, У. *UNIX: взаимодействие процессов*. СПб.: Питер, 2003.
22. Тодорова, М., *Езици за функционално и логическо програмиране, I ч.: Функционално програмиране*. Сиела, София, 2003.
23. Тодорова, М., *Обектно-ориентирано програмиране на базата на езика C++*, Сиела софт енд паблишинг АД, 2008.
24. Тодорова, М., *Програмиране на C++, I и II част*. Ciela, София, 2002.
25. Тодорова, М., *Структури от данни и програмиране на C++*, Сиела Норма АД, 2011.
26. Тодорова, М., Армянов, П., Николов, К., *Сборник от задачи по програмиране на C++: част втора, Обектно-ориентирано програмиране*, ТехноЛогика, 2008.
27. Тодорова, М., Армянов, П., Петкова, Д., Николов, К., *Сборник от задачи по програмиране на C++: част първа*, Увод в програмирането, ТехноЛогика, 2008.

28. Уирт, Н., *Алгоритми + структури от данни = програми*, BG soft group, София.
29. Abelson, H., & Sussman, G. J. *Structure and interpretation of computer programs*. The MIT Press, 1996.
30. Clements P., Bachmann F., Bass L., Garlan D., Ivers J., Little R., Merson P., Nord R., Stafford J., *Documenting Software Architectures: Views and Beyond*, Addison-Wesley Professional, 2010, ISBN: 0-321-55268-7.
31. Cormen, T., Ch. Leiserson, R. Rivest, C. *Stein, Introduction to Algorithms*, Third Edition, MIT Press, 2009.
32. Garcia-Molina, H., J. Ullman, J. Widom, *Database Systems: The Complete Book*, Prentice Hall, 2002.
33. Grimaldi, R., *Discrete and Combinatorial Mathematics: An Applied Introduction*, Pearson, 5 edition, 2003, ISBN-13: 978-0201726343.
34. Hopcroft, John E., and Jeffrey D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation*. First. Addison-Wesley.
35. Larry L. Peterson and Bruce S. Davie, *Computer Networks: A Systems Approach* Fifth Edition, © 2012 Elsevier, Inc.
36. Martin, J., *Introduction to Languages and the Theory of Computation*, McGraw-Hill, 4 edition, 2010, ISBN-13: 978-0073191461.
37. Nilsson, U., J. Maluszynski, *Logic, Programming and Prolog* (2nd ed.). John Willey & Sons, 1995. Електронно издание: http://www.ida.liu.se/~ulfni/lpp/.
38. Papadimitriou, Christos, and Harry Lewis. 1998. *Elements of the Theory of Computation*. Prentice-Hall.
39. Pressman R., Maxim B., *Software Engineering: A Practitioner's Approach*, McGraw-Hill, 8/e, 2015, ISBN: 0078022126.
40. Rosen, K., *Discrete Mathematics and Its Applications*, McGraw-Hill Education, 7 edition, 2012, ISBN 9780073383095.
41. Russell, S., P. Norvig. *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson, 2020.
42. Silberschatz, Galvin, and Gagne, *Operating System Concepts*, 7th Edition
43. Sipser, Michael. 2012. *Introduction to the Theory of Computation*. 3rd ed. Cengage Learning.
44. Sommerville I., *Software Engineering*, Addison-Wesley 10th ed., 2015.
45. Stallings, W., *Computer Organization and Architecture. Design for Performance*, Prentice Hall, 2000. http://www.williamstallings.com/COA5e.html.
46. Stevens W.R. *Advanced Programming in the UNIX Environment*, Addison-Wesley, Reading, Mass, 1992.
47. Stroustrup, B., *C++ Programming Lanquage*. Third Edition, Addison-Wesley, 1997.
48. Tanenbaum, A., *Modern Operating systems*, 2nd ed., Prentice Hall, 2002.
49. Tanenbaum, A., Bos, H., *Modern Operating Systems*, 4th ed. Pearson, 2014, (глави 10-12).
50. Tanenbaum, A., *Structured Computer Organization*. Prentice Hall, 2002.
51. Tannenbaum Andrew S., Wetherall David J., *Computer Networks*, 5th ed., Prentice Hall, 2011, http://libgen.org/book/index.php?md5=1990789686fa1463f09d2fb230d4301c.
52. Taylor R., Medvidovic N., Dashofy E *Software Architecture: Foundations, Theory, and Practice*, John Wiley & Sons, 2009, ISBN 0470167742.
53. Thompson, S. Haskell: *The Craft of Functional Programming* (2nd ed.). Addison-Wesley, 1999.