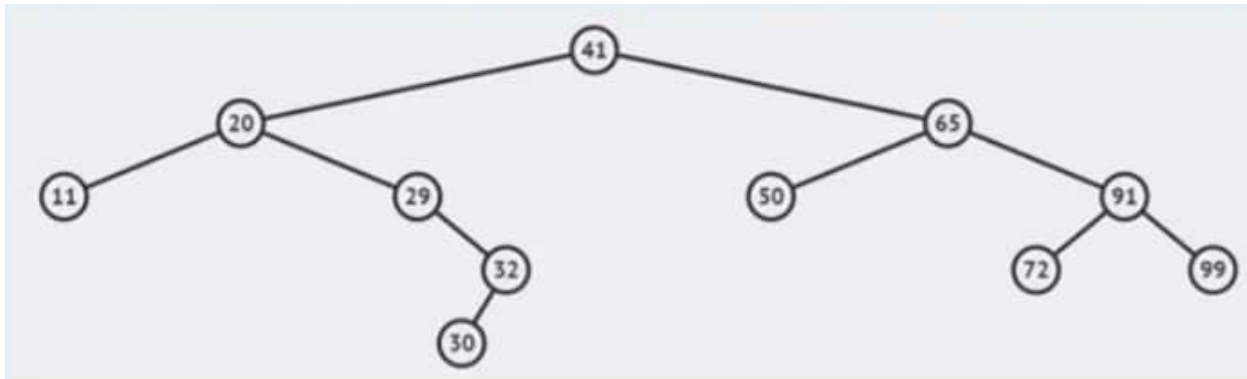


1. После операцијата вметнување елемент AVL дрвото се дебалансира. Која операција треба да се преземе за да стане правилно AVL дрво.

- Единечна ротација - десно
- *Двојна ротација - лево, десно*
- Единечна ротација - лево
- **Двојна ротација - десно, лево 98**

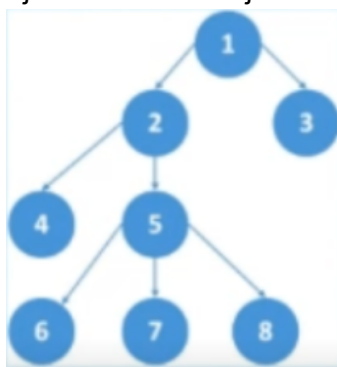


2. Замислете дека постојат две решенија на еден проблем. Првото решение има два последователни рекурзивни повика кои го користат скоро почетниот број на елементи. Второто решение ги изминува половина од елементите во циклус и потоа втората половина од елементите во обратен циклус. Кое од решенијата е поефикасно и зошто?

- Првото затоа што рекурзија е секогаш поефикасна од циклус
- **Второто затоа што го решава проблемот во линеарно време наместо во експоненцијално**
- Првото затоа што има експоненцијална комплексност за разлика од второто што има двојна линеарна комплексност.
- Првото, затоа што има логоратимска комплексност за разлика од второто кое има линеарна комплексност.

Првото има логоритамска второто линеарна

3. Која е висината на јазелот 1 на дрвото на сликата?



- Нема висина
- 1
- **3**
- 2

4. Со која техника на програмирање може ефикасно да се најде најдолга растечка подниза на дадена низа?

- Лакоми алгоритми
- **Динамичко програмирање**
- Техники со враќање наназад
- Груба сила

- 4) — 1 — 2--- 3
- 1 -- 2---3)
- 3) — 1 —> 2---4
- 2) — 1- --4 (a zosto e ova?) i mene me buni - koga ke se pretstavi matricata kako lista na sosdstvo i posle sekoe teme da se pretstavi koe so koe e povrzano, temeto 2 e povrznano so 1 i 4 a drugite se gresni

- комбинација на алчни алгоритми и груба сила
- Препопување на влезно множество на податоци
- **Паметење на пресметани резултати и генерирање на простор на резултати**
- рекурзија и брзи математички пресметки

- Имплицитно се имплементира со итерација
- Користи приоритет
- Може да се имплантира само со двојно поврзана листа
- **Имплицитно се имплементира со рекузија**

- 20, 47, 8, 15, 4, 9, 30, 40, 12, 17
- **8, 15, 20, 47, 4, 9, 30, 40, 12, 17**
- 4, 8, 9, 15, 20, 47, 12, 17, 30, 40
- 15, 20, 47, 4, 8, 9, 12, 30, 40, 17

Key Points

- Input : 20 47 15 8 9 4 40 30 12 17
- Pass 1 : (20 47) (8 15) (4 9) (30 40) (12 17)
- Pass 2 : 8, 15, 20, 47, 4, 9, 30, 40, 12, 17.

- **стек**
- приоритетна листа
- редица
- ниту една од понудените апстрактни податочни структури

10. Која од следниве карактеристики НЕ е недостаток при користење на низи?
- Постои можност за неискористеност на мемориски простор ако елементите во низата се помалку отколку а
11. Ако елементите "A", "B", "C" и "D" се вметнат во празна редица, кој ќе биде редоследот на вадење на елементите од редицата?
- ABCD
 - DCBA
12. Стекот (Магацинот) претставува еднодимензионална линеарна секвенца од елементи која работи по принципот:
- Последен-внесен-последен-изваден (last-in-last-out)
 - **последен-внесен-прв-изваден**
(last-in-first-<https://bbb-lb.finki.ukim.mk/playback/presentation/2.3/0c6ccaedfc3fb2126b9f93b160daa72f755b267-1633348039366?meetingId=0c6ccaedfc3fb2126b9f93b160daa72f755b267-1633348039366out>)
 - прв-внесен-последен-изваден (first-in-last-out)
 - прв-внесен-прв-изваден (first-in-first-out)
- недост
13. Додавањето на нови елементи и нивното вадење од Стекот (Магацинот):
- се врши од дното
 - зависи од имплементацијата на магацинот
 - **се врши од врвот**
 - зависи од пополнетоста на магацинот
- 14.
- подредени елементи (јазли) што содржат вредност и покажувачи кон произволен број јазли
 - **подредени елементи (јазли) што содржат вредност и покажувач кон следен јазел**
 - елементи (јазли) што содржат вредност и покажувач кон следен јазел (ova bi trebalo da e)(Ne e proveriti vo prezentacija smh)
 - подредени елементи (јазли) што содржат вредност и покажувач кон следната вредност
15. Што е недостаток на податочната структура Низа?
- има константно време на пристап до елемент преку неговиот индекс
 - **не може да се промени нејзината големина**
 - не се чуваат други помошни податоци во меморија (пример покажувачи)
 - времето на изминување на целата низа е брзо
16. Кој од следните искази НЕ е точен, кога станува збор за еднострано поврзана листа?
- Додавањето и бришењето на елементи е поедноставно отколку кај низа
 - еднострано поврзана листа не овозможува ефикасен пристап до произволен елемент
 - еднострано поврзаната листа не може да се наполни (освен ако се наполни меморијата)
 - **кај еднострано поврзана листа не е потребна дополнителна меморија за чување на покажувач**
17. Кога се вметнува елемент во редица, истиот се вметнува на ____ на редицата, со операцијата_____.
- почетокот (главата), dequeue
 - почетокот(главата), enqueue
 - крајот(опашката), dequeue
 - **крајот(опашката), enqueue**
18. Целта на робот за пакување е да смести предмети во пакети, така што вкупната вредност на предметите во пакетот се максимизира. Предметите не може да се делат и мора

целосно да ги собира во пакетот. Максималниот волумен на пакетот е 50 литри. Предметите се со волумен {15, 20, 30, 35} и вредности {90, 100, 120, 125} . Која е максималната вкупна вредност на предметите што ги собира во пакетот?

- **220=120+100**
- 200
- 215=90+125
- 190=90+100

19. Претпоставете дека имате монети од 1, 3 и 4 денари. Користејќи алчен алгоритам кој ќе ја избира најголемата монета која не е поголема од преостанатата сума, одговорете за која од следниве суми, алгоритмот нема да даде оптимален одговор?

- **6 (треба 3+3 а тој ќе врати 4+1+1)**
- 20 (4+4+4+4+4)
- 5 (4+1)
- 12 (4+4+4)

20. Која од следните техники ја користи Merge sort за имплементација на сортирањето?

- Враќање наназад (backtracking)
- Динамичко програмирање
- **Раздели и владеј**
- Алчен (greedy) алгоритам

21. Кога алгоритмот Quicksort има најлоши перформанси:

- Кога броевите во низата се рамномерно дистрибуирани во одреден опсег
- Кога имаме мал број на различни елементи
- Кога имаме голем број на различни елементи
- **Кога низата е скоро сортирана**

22. Која од следниве операции не е $O(1)$ за **сортирана** низа со елементи. Може да претпоставите дека елементите во низата се различни.

- **Избриши елемент // $O(n)$ е, има после шифтање на др елементи за да се пополни празното место**
- Најди го i -тиот најголем елемент
- Најди го i -тиот најмал елемент
- Сите наведени

23. Во имплементација на редица со еднострано поврзана листа, ако се чува само покажувачот кон front, која од следните операции е со најлошо време на извршување?

- **И додавање на елемент и бришење на цела редица**
- Додавање на елемент
- Бришење на цела редица
- Бришење на елемент

24. Која е сложеноста на алгоритмот:

- $O(N * M)$
- $O((N + M)/2)$
- $O((N * M)/(N + M))$
- **$O(N + M)$**

```

compute(M, N, x, y){
  int a = 0, b = 0;
  for (i = 0; i < N; i++) {
    a = a + y[i];
  }
  for (j = 0; j < M; j++) {
    b = b + x[j];
  }
}

```

25. Која е временската сложеност на алгоритмот со техника на груба сила (brute force) што се користи за решавање на проблемот со ранец?

- $O(n)$
- $O(n^3)$
- **$O(2^n)$**
- $O(n!)$

26. Кои од следниве тврдења се неточни за двојно поврзани листи?

- Потребно е повеќе мемориски простор за нив во споредба од еднострано поврзани листи
- Имплементација на двојно поврзана листа е полесно отколку на еднострано поврзана листа
- **Додавањето и бришењето на јазел е подолго**

27. Да се напише функцијата за бројот на извршени операции за дадениот псевдо код и да се пресмета асимптотската горна граница "големо о" за истиот код:

- $f(n)=2n*n+4$, $O(n*n)$
- $f(n) = 5n+8$
- $f(n) = 3n+6$, $O(3n)$
- $f(n)=2n+8$. $O(n)$

```

simple_computer(x, n)
{
  int sum=0;
  for (int i=0; i<n; i++)
    sum+=2*i*x[i];
  return sum;
}

```

28. Кој од следните алгоритми има еднаква најдобра, најлоша и просечна временска комплексност?

- **Heap sort** (+ Merge sort, Radix, Counting)
- Insertion sort
- Bubble sort
- Quick sort

29. Колкава е сложеноста на алгоритмот со кој се прави спојување на две сортирани листи со големина m и n во сортирана листа со големина $m + n$?

- $O(\log m + \log n)$
- $O(n)$
- $O(m)$
- **$O(m+n)$**

30. Дадени се следните функции:

push(): push (додади) елемент на стекот

pop() : pop (избриши) го елементот од врвот на стекот

top() : врати го елементот кој е на врвот на стекот

Кој ќе биде излезот откако ќе се изврши следната низа на операции:

push(20);

```

push(4);
top();
pop();
pop();
push(5);
top();

```

- 5
- 4
- 20
- stack underflow

31. Што значи кога велиме дека алгоритмот X е асимптотски поефикасен од Y?
- **X секогаш ќе биде подобар избор за големи влезови** (ова е точниот одговор)
 - Y секогаш ќе биде подобар избор за мали влезови
 - X секогаш ќе биде подобар избор за сите влезови
 - X секогаш ќе биде подобар избор за мали влезови

32. Која е просторната (мемориската) сложеност за бришење на поврзана листа?
- $O(\log n)$
 - Или $O(1)$ или $O(n)$
 - **$O(1)$**
 - $O(n)$

33. За временската комплексност на пребарување во хеш табела со СВНТ и ОВНТ важи:
- Комплексноста во најлош случај при пребарување со ОВНТ е поголема
 - **Во најлош случај, пребарување со СВНТ и ОВНТ имаат иста комплексност**
 - Не може да се спореди комплексноста при пребарување во најлош случај меѓу СВНТ и ОВНТ
 - Комплексноста во најлош случај при пребарување со СВНТ е поголема

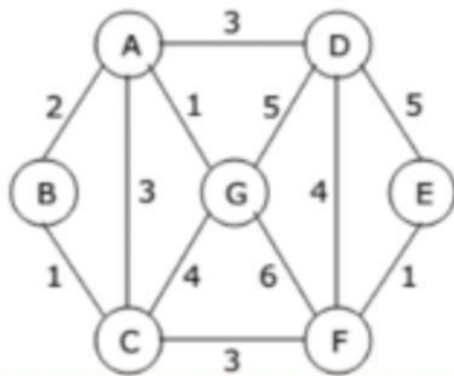
34. Клучевите 12, 18, 13, 2, 3, 23, 5 и 15 се внесуваат во иницијално празна хеш табела со должина 10 која користи ОВНТ со хеш функција $h(k) = k \bmod 10$ со $step(k) = k \bmod 5$. Која е резултантната хеш табела?

A.	<table> <tr><td>0</td><td>15</td></tr> <tr><td>1</td><td></td></tr> <tr><td>2</td><td>12</td></tr> <tr><td>3</td><td>13</td></tr> <tr><td>4</td><td>2</td></tr> <tr><td>5</td><td>5</td></tr> <tr><td>6</td><td>3</td></tr> <tr><td>7</td><td></td></tr> <tr><td>8</td><td>18</td></tr> <tr><td>9</td><td>23</td></tr> </table>	0	15	1		2	12	3	13	4	2	5	5	6	3	7		8	18	9	23
0	15																				
1																					
2	12																				
3	13																				
4	2																				
5	5																				
6	3																				
7																					
8	18																				
9	23																				
B.	<table> <tr><td>0</td><td>5</td></tr> <tr><td>1</td><td></td></tr> <tr><td>2</td><td>12</td></tr> <tr><td>3</td><td>3</td></tr> <tr><td>4</td><td>2</td></tr> <tr><td>5</td><td>15</td></tr> <tr><td>6</td><td>13</td></tr> <tr><td>7</td><td>23</td></tr> <tr><td>8</td><td>18</td></tr> <tr><td>9</td><td></td></tr> </table>	0	5	1		2	12	3	3	4	2	5	15	6	13	7	23	8	18	9	
0	5																				
1																					
2	12																				
3	3																				
4	2																				
5	15																				
6	13																				
7	23																				
8	18																				
9																					
C.	<table> <tr><td>0</td><td></td></tr> <tr><td>1</td><td></td></tr> <tr><td>2</td><td>12</td></tr> <tr><td>3</td><td>13</td></tr> <tr><td>4</td><td>2</td></tr> <tr><td>5</td><td>3</td></tr> <tr><td>6</td><td>23</td></tr> <tr><td>7</td><td>5</td></tr> <tr><td>8</td><td>18</td></tr> <tr><td>9</td><td>15</td></tr> </table>	0		1		2	12	3	13	4	2	5	3	6	23	7	5	8	18	9	15
0																					
1																					
2	12																				
3	13																				
4	2																				
5	3																				
6	23																				
7	5																				
8	18																				
9	15																				
D.	<table> <tr><td>0</td><td>5</td></tr> <tr><td>1</td><td></td></tr> <tr><td>2</td><td>12</td></tr> <tr><td>3</td><td>13</td></tr> <tr><td>4</td><td>15</td></tr> <tr><td>5</td><td>2</td></tr> <tr><td>6</td><td></td></tr> <tr><td>7</td><td>3</td></tr> <tr><td>8</td><td>18</td></tr> <tr><td>9</td><td>23</td></tr> </table>	0	5	1		2	12	3	13	4	15	5	2	6		7	3	8	18	9	23
0	5																				
1																					
2	12																				
3	13																				
4	15																				
5	2																				
6																					
7	3																				
8	18																				
9	23																				

- **A// zaso e 15 na 0 zaradi step? $15 \bmod 5 = 0$ зашто 0 е празен а 5 е зафатен со 5 k**
- B
- C
- D

35. Еднократната ротација кај AVL дрво може да се изврши врз
- коренот на дрвото
 - јазелот со најмала вредност од дрвото
 - било кој лист од дрвото
 - **било кој јазел од дрвото**

36. За дадениот граф, почнувајќи од темето A, кој е точниот редослед на ребра кои се додаваат во минималното распнувачко стебло со примена на алгоритмот на Прим?



- (A, G) — (A, B) — (A, C) — (A, D) — (A, D) — (C, F)
- (A, G) — (G, C) — (C, B) — (C, F) — (F, E) — (E, D) -
- (A, G) — (B, C) — (E, F) — (A, B) — (C, F) — (D, E)
- **(A, G) — (A, B) — (B, C) — (A, D) — (C, F) — (F, E) -**

37. Кај AVL дрво длабочината на дрвото (а со тоа и комплексноста на најчестите операции) е од редот (+ BST, B-tree)

- $O(n \log n)$
- **$O(\log n)$**
- $O(C)$
- $O(n)$

37. Изминување на граф е различно од изминување на дрва, бидејќи:

- **графовите може да имаат циклуси**
- дрвата имаат корен
- дрвата не се поврзани
- ниту еден одговор не е точен

38. Доколку бинарното дрво е пребарувачко, тогаш:

- Јазелот со најголем клуч е јазелот до кој се доаѓа доколку од коренот се оди само по левите врски се додека не се дојде до листот на дрвото.
- **Јазелот со најмал клуч е јазелот до кој се доаѓа доколку од коренот се оди само по левите врски се додека не се дојде до листот на дрвото.**
- Јазелот со единствен клуч е јазелот до кој се доаѓа доколку од коренот се оди само по левите врски се додека не се дојде до листот на дрвото.
- Јазелот со нулев клуч е јазелот до кој се доаѓа доколку од коренот се оди само по левите врски се додека не се дојде до листот на дрвото.

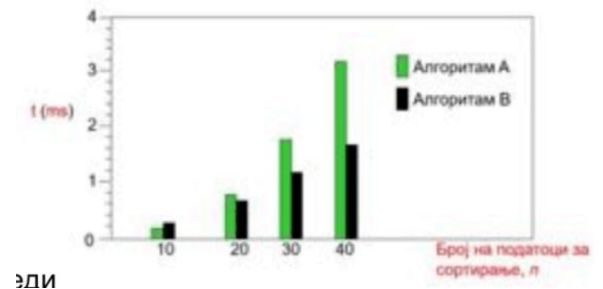
39. При бришење на јазел со две деца од пребарувачко дрво,

- клучот на јазелот што треба да се избрише се заменува со било кој клуч од неговото лево поддрво
- клучот на јазелот што треба да се избрише се заменува со најголемиот клуч од неговото лево поддрво
- Клучот на јазелот што треба да се избрише се заменува со најмалиот клуч од неговото лево поддрво
- клучот на јазелот што треба да се избрише се заменува со најголемиот клуч од неговото десно поддрво

Клучот на јазелот што треба да се избрише се заменува со најмалиот клуч од неговото десно поддрво.(neli e vaka????) DA

40. Кој алгоритам е побрз А или В (за дадените времиња на извршување на сликата)?

- Алгоритамот А
- **Алгоритамот В**
- Исто се брзи
- Не може да се спореди



41. Колку е $O()$ за дадениот код?

```
int maximum(int a[], int n)
{
    int i, max = a[0];
    for (i = 1; i < n; i++)
    {
        if (a[i] > max)
            max = a[i];
    }
    return max;
}
```

- $O(1)$
- $O(a*n)$
- **$O(n)$**
- $O(n^2)$

42. Колку е $O()$ за дадениот код?

```
for (i=0; i<n; i++)
    a[i] = i;
for (i=0; i<n; i++)
    for (j=0; j<n; j++)
        a[i] += a[j]+j*2;
```

- $O(n^3)$
- $O(i)$
- $O(n)$
- **$O(n^2)$**

43. Што од наведеното е точно?

- AVL дрвото има подредено (сортирано) преордер изминување.
- AVL дрвото има поврзани терминални јазли.
- **AVL дрвото мора да биде балансирано.**
- AVL дрвото има константен степен на своите јазли.

44. Во кој случај е препорачливо да се користи Insertion sort за сортирање на низа од броеви

?

- Кога броевите во низата се рамномерно дистрибуирани во одреден опсег
- **Кога низата е скоро сортирана //**
- Кога имаме голем број на различни елементи
- Кога имаме мал број на различни елементи

Time Complexity

Best Case:

- **Selection sort:** The best-case complexity is $O(N^2)$ as to find the minimum element at every iteration, we will have to traverse the entire unsorted array.
- **Bubble sort:** The best-case complexity is $O(N)$. It happens when we have an already sorted array, as in that case, we will stop the procedure after a single pass.
- **Insertion sort:** The best-case complexity is $O(N)$. It occurs when we have a sorted array; as in that case, each element has already been placed at its correct position, and no swap operation will be required.

45. При имплементација на стек со поврзана листа, додавањето и вадењето на елемент се врши

- **На почетокот на листата** (проверено точно!)
- На крајот на листата
- На произволна локација во листата
- На средината на листата

46. Кој е **излезот** после дадената листа на операции:

push(5)

push(8)

pop

push(2)

push(5)

pop

pop

pop

push(1)

pop

- 8 5 5 2 1
- **8 5 2 5 1**
- 8 2 5 5 1
- 8 1 2 5 5

47. Кај двојно поврзана листа, додавањето на нов елемент после веќе покажан елемент има сложеност:

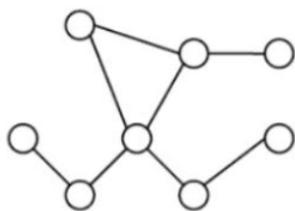
- $O(n \cdot \log(n))$
- $O(n)$
- $O(n \cdot n)$
- **ниту едно од понудените - *treba da e O(1)***

48. Во нетежински ненасочен поврзан граф, најкраткиот пат од јазол S до секој друг јазол, најефикасно може да се пронајде, во смисол на временска комплексност, со:

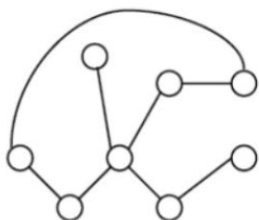
- Алгоритмот на Крускал
- Длабочинско изминување на графот (DFS)
- **Ширинско изминување на графот (BFS)**
- Алгоритмот на Дијкстра

49. Which of the following graphs is isomorphic to:



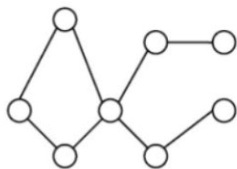


-
-



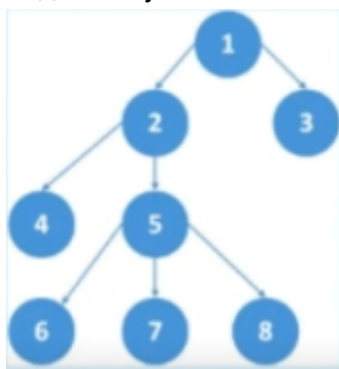
-

ova |
v



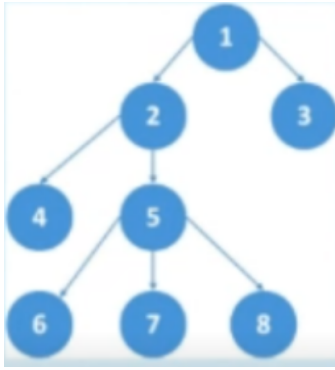
-

50. Доколку дрвото на сликата се трансформира во бинарно дрво тогаш кој јазол ќе биде десно дете на јазолот 2?



- 5
- 1
- нема да има такво дете
- **3 tocen odg**

51. Доколку дрвото на сликата се трансформира во бинарно дрво тогаш кој јазол ќе биде десно дете на јазолот 5?

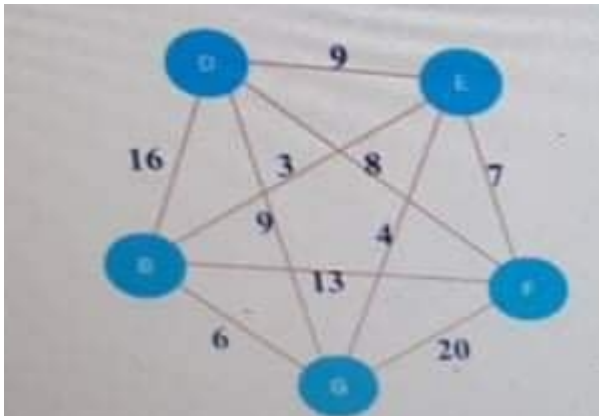


- 8
-
- 7
- **нема да има такво дете**
- 6

52. Податочната структура heap, може да се имплементира со помош на еднодимензионална низа поради фактот што: // Since a Binary Heap is a Complete Binary Tree, it can be easily represented as an array and array-based representation is space-efficient. ова од интернет, ваљда првото? Тоа е prvoto

- **е комплетно дрво**
- е бинарно дрво
- е слабо подредено бинарно дрво
- ги задоволува условите на heap дрво

53. Земете го предвид следниот граф. Користејќи го алгоритмот на Крускал, кое ребро треба да биде избрано прво?



- BG
- DE
- GF
- **BE- REBRO SO NAJMALA TEZINA (vo nas slucaj toa e 3)**

54. Наоѓање јазол во бинарно пребарувачко дрво вклучува одење од јазол на јазол и прашување:

- кој јазол-лист сакаме да го достигнеме
- на кое ниво сме
- **дали јазолот што го бараме е поголем/помал од моменталниот јазол**
- дали јазолот што го бараме е поголем/помал од десното или левото дете

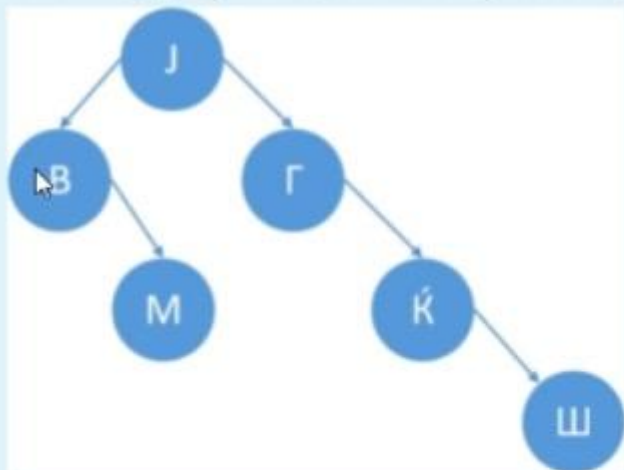
55. Секое B-дрво од ред p , колку клучеви може да има во внатрешен јазел?

- n
- $n-1$
- 2
- 1

56. Нацртајте ја хеш табела со затворени кофички со должина 9 по вметнување на елементите 10, 35, 18, 2, 19, 26, 9, 28. Како хеш функција се користи $h(x) = x \bmod 9$. Колку елементи ќе има во кофичките со реден број 1 и 2?

- 2 и 2
- **3 и 1**
- 3 и 0
- 2 и 0

Како ќе изгледа резултатот од inorder изминување на следното дрво:



Select one:

- ☐ a. M B Ш К Г J
- ☐ b. J B M Г К Ш
- ☐ c. B M J Г К Ш
- ☐ d. M B J Г К Ш

Како ќе изгледа резултатот од inorder изминувањето на следното дрво:

In order LEFT->ROOT->RIGHT

- M B Ш К Г J ova e za postorder
- J B M Г К Ш ova e za preorder
- **B M J Г К Ш**
- M B J Г К Ш

57. Нека е дадена репрезентацијата на графот со матрица на соседство $A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$, каде што редоследот на колоните и редиците е 1, 2, 3, 4, соодветно. Ако се примени пребарување по длабочина почнувајќи од темето 2 тогаш кој од следните исписи на темињата е можен.

- **2 4 3**
- 1 2 4 3
- 2 1 3 4
- 2 3 1 4

58. При пребарување во ОБНТ, алгоритмот застанува кога:

- Кофичката b е никогаш-зафатена
- Кофичката б е зафатена
- кофичката b е претходно-зафатена
- **Кофичката b е никогаш-зафатена или кофичката b е зафатена со еднаков клуч**

59. Најдобри перформанси при користење на ОБНТ се добиваат со:

- Една хеш функција и степ функција со чекор 1
- Една хеш функција и една степ функција со произволен чекор
- **Една хеш функција и степ функција со двојно хеширање**
- Една хеш функција

60. Нека е дадена следната хеш табела која користи отворени кофички. Хеш функцијата е $h(x) = x \bmod 9$, а додека пак чекорот е $\text{step}(k) = 1$. По кој редослед биле додавани елементите во хеш табелата? Постојат повеќе точни одговори.

0	1	2	3	4	5	6	7	8
9	18		12	3	14	4	21	

12 14 3 9 4 18 21

- 9 14 4 18 12 3 21
- **9 12 14 3 4 21 18**
- 12 3 14 18 4 9 21
- 12 9 18 3 14 21 4**step е за koga е зафатена кофичката со треба да ставаме у неа и ставаме у таа според stepot, HEAt.e у 1vata после зафатената во случајов.

61. При бришење јазел од hear дрво потребно е да се направат следните акции:

- **Бришење на коренот на дрвото; Прилагодување на hear дрвото**
- Бришење на коренот на дрвото и на негово место поставување на најдесниот јазел
- Бришење на најдесниот јазел-лист
- Бришење на најмалиот елемент во hear дрвото

62. Кога би требало во една хеш табела да Сместиме 1000 податоци кои претставуваат имиња на артикли со соодветен бар код, како ќе ја креираме мапата (клуч, вредност)?

- (име-артикл, бар-код)
- **(бар-код, име-артикл)**
- (име-артикл, име-артикл+бар-код)
- (бар-код, име-артикл+бар-код)

63. Која од следните би била најдобар избор за хеш функција?

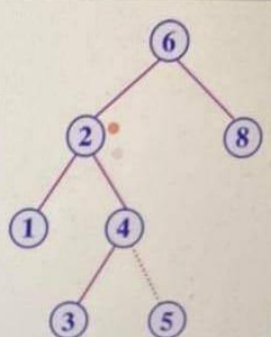
- Нема воопшто колизии, а сложеноста е непозната
- Има многу ретки колизии, а сложеноста е $O(\log N)$
- Нема воопшто колизии, а сложеноста е $O(N)$
- **Има ретки колизии, а сложеноста е $O(1) \approx O(1)$**

64. При работа со СВНТ вметнување елемент во листата на кофичката со клуч $\text{hash}(\text{key})$ се прави со која операција од SLL?

- **insertFirst**
- insertLast //
- insertBefore
- insertAfter

Прашање 3

☐ Нека е дадено следното BST. После бришење на ел. 2 дрвото ќе изгледа вака:



A. 6-2-8-1-4-3-5
 B. 6-1-8-4-3-5
 C. 6-5-8-1-4-3
 D. 6-3-8-1-4-5

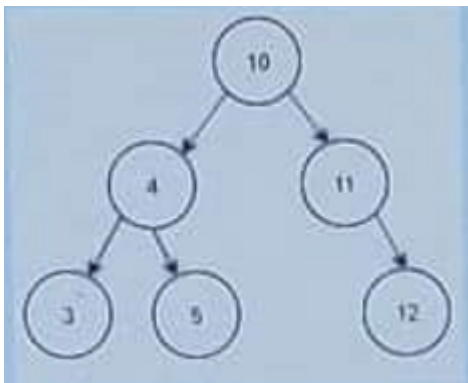
65. u

- 6-2-8-1-4-3-5
- 6-1-8-4-3-5
- 6-5-8-1-4-3
- **6-3-8-1-4-5** //najleviot jazol na desnoto dete

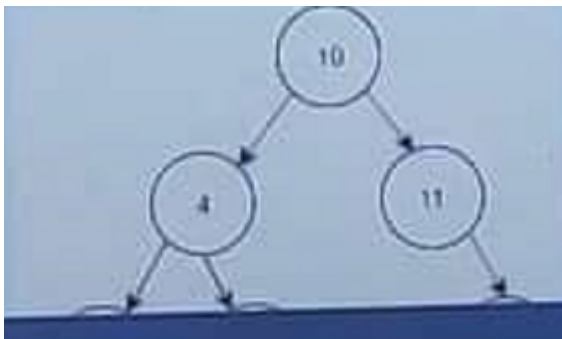
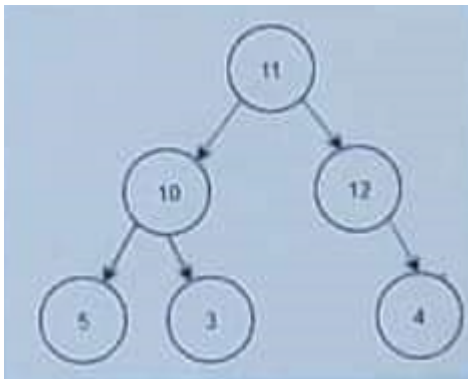
66. Бинарно пребарувачко дрво е:

- Комплетно бинарно дрво во кое секој јазел што претставува корен на свое поддрво има минимална вредност
- Полно бинарно дрво во кое елементите во левото поддрво се помали или еднакви на коренот, а елементите во десното поддрво се поголеми од коренот
- **Бинарно дрво во кое елементите во левото поддрво се строго помали, а оние во десното поддрво строго поголеми од коренот**

67. Да се конструира бинарно пребарувачко дрво, ако при изминување со преордер начин се добива следнава секвенца: 10, 4, 3, 5, 11, 12.



ova



68. Кој алгоритам може да се искористи за пресметка на рата на кредит?

- DFS
- **Бинарно пребарување**
- Сортирање со спојување
- BFS

69. Доколку при разгледувањето на просторот на решението на даден проблем забележувате дека одредени негови делови не треба да се разгледуваат, станува збор за користење на техниката

- Разделување

- Мешање
- Претворање
- **Поткастрување (prune the tree)**

70. Празна (ретка, sparse) матрица е

- матрица со голем број на нули што би требало да се компресираат
- матрица со нули што треба да се компресираат од аспект на индексите што ги користи
- матрица со голем број на нулеви вредности што треба да се компресираат од аспект на индексите што ги користи
- **матрица со голем број на нулеви вредности што треба да се компресираат мемориски**

71. Што од наведеното е точно?

- Бројот на нетреминални кај комплетно дрво со степен три не е поврзан со бројот на терминални јазли кај дрва со степен три.
- **Бројот на нетреминални јазли кај комплетно дрво со степен три е секогаш помал од бројот на терминални јазли.**
- Бројот на нетреминални јазли кај комплетно дрво со степен три е секогаш поголем или еднаков од бројот на терминални јазли.
- Бројот на нетреминални јазли кај комплетно дрво со степен три е секогаш еднаков на бројот на терминални јазли.

72. Кој алгоритам е најдобар при скоро sortirana lista (максимално 1 или 2 елементи не се на своите места)?

- Bubble sort
- Сортирање со спојување (merge sort)
- **Сортирање со вметнување (insertion sort)**
- Брзо сортирање (Quick sort)

73. За низа од n броеви во интервалот од 1 до n^6 , кој алгоритам Може да се употреби за сортирање на низата во линеарно време? [// Radix Sort е точниот одговор. Ова прашање падна на термин 2 во јуни 2022](#)

- Не е возможно да се сортира во линеарно време
- **Radix Sort**
- Insert Sort
- Quick Sort

74. Што може да се моделира со графови?

- Протеински врски
- Шаховска табла
- Уште некој пример
- **Сите претходни**

75. Which algorithm can be used to efficiently calculate a loan installment amount?

- Merge sort
- BFS
- **Binary search**
- DFS

76. Which sorting algorithm will take least time when all elements of input array are identical? Consider typical implementations of sorting algorithms.

- SELECTION SORT
- HEAP SORT
- **INSERTION SORT**
- MERGE SORT

77. Кој од следниве алгоритми е најефикасен ако опсегот на броевите во низата не е значително поголем од должината на низата?

Counting sort is most efficient if the range of input values is not greater than the number of values to be sorted.

- Radix Sort
- Selection sort

- Quick Sort
- **Counting sort**

78. Потребно е да имплементирате хеш табела за складирање на 1000 податоци за студенти, чии клучеви се во форма XXVIII, каде XX е година на запишување (последни 2 цифри), Y е насока - една од 7те и е реден број во рамките на насоката и годината. Која опција за хеширање најмногу би одговарала?(0,50-0,75)

- 1499 кофички, хеш функција која ги користи сите цифри при пресметка на клучот
- 1199 кофички, хеш функција која ги користи првите 4 цифри за пресметка на клучот
- 9997 кофички, хеш функција која ги користи првите 2 и последните 2 цифри за пресметка на клуч
- **1499 кофички, хеш функција која ги користи последните 4 цифри во пресметка на клучот**

“

Нека $m = 10000$, $\text{hash}(id) = \text{последните 4 цифри од } id$. + Добар број на кофички. Load factor $6000/10000 = 0.6$. – Лоша дистрибуција. Скоро сите id -а завршуваат со 0000...1500.

“ SPOred презентација ќе е првото

79. Архетипот на алчни алгоритми

- наоѓа точно решение за одреден тип на проблеми
- **наоѓа најбрзо решение за одреден тип на проблеми**
- секогаш наоѓа точно решение за одреден тип на проблеми
- наоѓа најбрзо решение за било кој тип на проблеми

80. Дадени предмети се претставени како подредени парови (тежина, вредност) ((10,60), (7, 28), (4,20), (2,24)). Капацитетот на ранецот е 7. Да се најде максималната вредност препоставувајќи дека има само еден од секој предмет и предметите се деливи ().

//дали може постапката да се напише како дојде до 54? **mislam deka se gleda odnosot koj najmnogu vredi da se zeme i toa e $24/2=12$, znaci go stavame toj so teжина 2 vrednost 24, pa ni ostanuva uste 5 kapacitet i sleden najmnogu vredi $60/10=6$ i od nego zemame pola odnosno teжина 5 vrednost $30 = 24+30=54$** , јасно фала (imate aud od nenad so matrici odnosno resenieto so dinamicko prog)

- 44
- 28
- 48
- **54**

81. Замислете дека постојат две решенија на еден проблем. Првото решение има два последователни рекурзивни повика кои го преполовуваат бројот на почетни елементи. Второто решение ги изменува половина од елементите во циклус. Кое од решенијата е поефикасно и зошто?

- Второто затоа што го решава проблемот само со половина од елементите, додека првото непотребно два пати ги третира елементите
- **Првото, затоа што има логаритамска комплексност за разлика од второто кое има линеарна комплексност ? -> DA**
- Првото затоа што има логаритамска комплексност за разлика од второто што има половично линеарна комплексност- не е ова? Ne postoji polovicna linearna kompleksnost
- Првото затоа што рекурзија е секогаш поефикасна од циклус

82. Која е комплексноста за внесување на елемент во приоритетна редица во најлош случај?

- $O(n^2)$
- $O(n)$
- $O(n \cdot \log n)$
- **$O(\log n)$**

83. Која е просторната (мемориската) сложеност за бришење на поврзана листа?

- $O(n)$
- или $O(1)$ или $O(n)$
- **$O(1)$**
- $O(\log n)$

84. За дадена низа $a[\text{left}...\text{right}]$, во кој алгоритам се сретнува следната инструкција (во псевдојасик): копирај $a[x+1...\text{right}]$ во $a[x..\text{right}-1]$

- а. **Бришење елемент $a[x]$**
- б. Ниту еден од понудените одговори
- с. Додавање елемент $a[x]$
- д. Копирање една низа во друга

85. Замислете дека постојат две решенија на еден проблем. Првото решение има рекурзивен повик со за еден намален број на почетни елементи. Второто решение има два вгнездени циклуси со сите елементи. Кое од решенијата е поефикасно и зошто?

- а. Второто затоа што има квадратна комплексност за разлика од првото кое има експоненцијална комплексност.
- б. Првото затоа што има логаритамска комплексност за разлика од второто што има квадратна комплексност.
- **с. Првото, затоа што има линеарна комплексност за разлика од второто кое има квадратна комплексност.**
- д. Второто затоа што циклуси се секогаш поефикасна од рекурзија.

86. Која е сложеноста на имплементација на операцијата додавање, а која на вадење елемент од приоритетна редица имплементирана со подредена низа?

- а. $O(1) - O(1)$
- **б. $O(N) - O(1)$ ---**
- с. $O(N) - O(N)$
- д. $O(1) - O(N)$

87. При имплементација на редица со низа, елементите се наоѓаат (е-низа, f-глава, r-опаш)

* помеѓу:

- а. **$e[f...r-1]$ воа нели е за низа а под d е за циклична низа?, и јас ова мислам дека е +1**
- б. $e[f...maxlen-1]$ i $e[0...r-1]$
- с. $e[0...maxlen-1]$
- **д. $e[f...r-1]$ ili $e[f...maxlen-1]$ i $e[0...r-1]$ — ова е за циклична!!!**

ChatGPT вели дека е точно под а.

Кој од следните операции се изведува поефикасно кај DLL од колку кај SLL

- а. вметнување на теме после теме со дадена локација
- б. изминување на листата за процесијање на секое теме
- **с. бришење на теме со дадена локација ---**
- д. пребарување зададено теме кај несортирана листа

88. Кое е точното за подредување на алгоритми во однос на нивната сложеност во најдобар случај.

- а. merge s. > selection s. > quick s. > insertion s
- **б. insertion s. < quick s. < merge.s < selection s ---**
- с. merge s.> quick s. > selection s. > insertion s
- д. merge s.> quick s. > insertion s. > selection s

Time Complexities of Sorting Algorithms:

Algorithm	Best	Average	Worst
Quick Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n^2)$
Bubble Sort	$\Omega(n)$	$\Theta(n^2)$	$O(n^2)$
Merge Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$
Insertion Sort	$\Omega(n)$	$\Theta(n^2)$	$O(n^2)$
Selection Sort	$\Omega(n^2)$	$\Theta(n^2)$	$O(n^2)$
Heap Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$
Radix Sort	$\Omega(nk)$	$\Theta(nk)$	$O(nk)$
Bucket Sort	$\Omega(n+k)$	$\Theta(n+k)$	$O(n^2)$

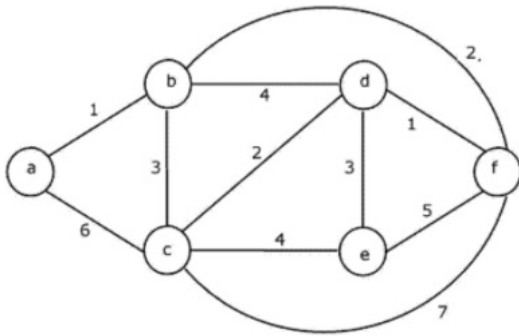
89. Дадени предмети се претставени како подредени парови {тежина, вредност}: $\{\{10,50\},\{7,28\},\{4,20\},\{2,24\}\}$. Капацитетот на ранецот е 13. Да се најде максималната вредност препоставувајќи дека има само еден од секој предмет и предметите не $c0/1$ knapsack).

- a. 48
- b. **74**
- c. 52
- d. 72

90. Замислете дека постојат две решенија на еден проблем. Првото решение има два последователни рекурзивни повика кои го користат скоро почетниот број на елементи. Второто решение ги изминува половина од елементите во циклус и потоа втората половина од елементите во обратен циклус. Кое од решенијата е поефикасно и зошто?

- **Второто затоа што го решава проблемот во линеарно време наместо во експоненцијално**
- Првото затоа што има логаритамска комплексност за разлика од второто што има линеарна комплексност.
- Првото, затоа што има експоненцијална комплексност за разлика од второто кое има двојна линеарна комплексност.
- Првото затоа што рекурзија е секогаш поефикасна од циклус.

91. Кој од следниве **НЕ** може да биде редослед на додадени ребра кога се прави минимално распнувачко дрво користејќи го алгоритмот на Крускал?



- **(d-f),(a-b),(b-f),(d-e),(d-c)/1 1 2 3 2**
- (a-b),(d-f),(d-c),(b-f),(d-e)
- (d-f),(a-b),(d-c),(b-f),(d-e)
- (a-b),(d-f),(b-f),(d-c),(d-e)

92. Со која податочна структура најефикасно се имплементира Приоритетна листа? Да се претпостави дека бројот на операциите вметнување и читање на моменталниот елемент со највисок приоритет, како и отстранување на елементот со највисок приоритет е скоро ист.

- несортирана низа
- поврзана листа
- **heap дрво**
- сортирана низа

93. Нека е дадена следната хеш табела која користи отворени кофички (ОВНТ). Хеш функцијата е $h(x) = X \bmod 9$, а додека пак чекорот е $step(k) = 1$. Нека претпоставиме дека се бришат елементите 14 и 21 од хеш табелета, а потоа се повикува внесување на елементите 13 и 23. Во кои кофички ќе бидат сместени елементите 13 и 23?

0	1	2	3	4	5	6	7	8
9	18		12	3	14	4	21	

- **5 i 7**
- 5 i 8,
- 8 i 2
- 7 i 8

94. Со која од наведените структури може лесно да се имплементира пронаоѓање на најблиска точка од интерес до тековната локација на мапа?

- Binary tree
- **Quad tree**
- A* tree
- B* tree

95. (juni 2022 termin 1)

Question 1

Not yet answered

Marked out of 1.00

Flag question

What does it mean when we say that an algorithm X is asymptotically more efficient than Y?

Select one:

- ☐ a. X will always be a better choice for all inputs
- ☐ b. X will always be a better choice for large inputs
- ☐ c. X will always be a better choice for small inputs
- ☐ d. Y will always be a better choice for small inputs

96.

Question 2

Not yet answered

Marked out of 1.00

Flag question

Which of the following is false about a doubly linked list?

Select one:

- ☐ a. It requires more space than a singly linked list
- ☐ b. Implementing a doubly linked list is easier than singly linked list
- ☐ c. The insertion and deletion of a node take a bit longer
- ☐ d. We can navigate in both the directions

C

97.

Question 3

Not yet answered

Marked out of 1.00

Flag question

The greedy algorithm pattern:

Select one:

- ☐ a. finds correct solution for some type of algorithmic problems.
- ☐ b. finds the fastest solution for any type of algorithmic problems.
- ☐ c. always finds correct solution for any type of algorithmic problems.
- ☐ d. finds the fastest solution for some type of algorithmic problems.

D

a? b? ц? d? a? b? aбe ц бe

98.

Question 5

Not yet answered

Marked out of 2.00

Flag question

Here is an array which has just been partitioned by the first step of quicksort:

3, 0, 2, 4, 5, 8, 7, 6, 9

Which of these elements could be the pivot?

Select one or more:

- ☐ 4
- ☐ 5
- ☐ 8
- ☐ 2

B i 4 moze isto neli???

99.

Question 6

Not yet answered

Marked out of 1.00

Flag question

You are tasked to implement a hash table for 1000 students with IDs in the format XXYZZZZ, where XX are the last two digits of the year of enrollment, Y is the one of the 7 study programs and ZZZZ is sequence number within the year and the program. Which hashing option would you select?

Select one:

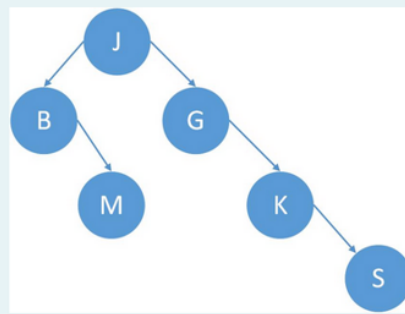
- ☐ a. 14999 buckets, with all six digits used to calculate the key
- ☐ b. 9997 buckets, with first 2 and last 2 digits used to calculate the key
- ☐ c. 1499 buckets, with the last four digits used to calculate the key
- ☐ d. 1199 buckets, with the first four digits used to calculate the key

100.

C

Question 7
Not yet answered
Marked out of 1.00
Flag question

What will be the result of the postorder traversal for the binary tree in the figure?



Select one:

- ☐ a. J B M G K S
- ☐ b. M B S K G J
- ☐ c. B M J G K S
- ☐ d. M B J G K S

101.

102.

Question 8
Not yet answered
Marked out of 2.00
Flag question

Let a graph be represented using an adjacency matrix: $A = [0 \ 1 \ 0 \ 0; 1 \ 0 \ 0 \ 1; 1 \ 0 \ 0 \ 1; 0 \ 1 \ 1 \ 0]$, where the order of columns and rows is 1, 2, 3, 4. If the graph is represented by a list of neighbors, which of the following is true?

Select one:

- ☐ a. $1 \rightarrow 2 \rightarrow 3$
- ☐ b. $3 \rightarrow 1 \rightarrow 2 \rightarrow 4$
- ☐ c. $2 \rightarrow 1 \rightarrow 4$
- ☐ d. $4 \rightarrow 1 \rightarrow 2 \rightarrow 3$

103.

C ne e B? "c" e točno,

Question 2
Not yet answered
Marked out of 1.00
Flag question

Што се случува кога top-down пристап на динамичко програмирање ќе се примени на било кој проблем?

Select one:

- ☐ a. Се зголемува мемориската комплексност, но се намалува временската комплексност
- ☐ b. Се зголемува временската комплексност, но се намалува мемориската комплексност
- ☐ c. Се намалуваат и мемориската и временската комплексност
- ☐ d. Се зголемуваат и мемориската и временската комплексност

^ A

4. Кое од следните тврдења е точно за тополошкото сортирање?

Select one:

- a. Ребрата кои можат да се вметнат во предходно изграденото минимално стебло се проверуваат според неопаѓачкиот редослед на нивните тежини
- b. На почеток ќе се избере минимално почетно ребро, кон него се додаваат ребра од сортираното множество на ребра
- c. Во секоја итерација се разгледуваат сите ребра кои не припаѓаат на веќе изграденото стебло, но излегуваат од темиња што припаѓаат на него
- d. На почеток се избира едно теме кое нема предходник, кое се „печати“ и отстранува од графот заедно со сите ребра кои излегуваат од него

D

Кај В-дрво сите податоци се сместени во:

- ЛИСЛОВИТЕ**
- јазлите на дрвото
- во индексите

Ако во едно дрво редоследот на поддрвата е многу важен, како се нарекува? -подредено дрво

1. .Што е редица? - Сложеност на пристапување на елемент според индекс во низа.
2. Кога е најдобро да се користи динамичко програмирање.
3. Со што најдобро би се превртеле елементи? - Стек

4. За адресирање презимиња се користи:-В дрво. (Хеш-табела ?)

5. Кога е граф прост? - Кога нема циклуси и паралелни ребра.
6. .Дали ако нема место во CD тогаш со алчен би се добиле оптимални резултати? -Не.
7. Сложеноста на алгоритми се мери со? -Ниту еден од понудените одговори.
8. Кај приоритетна редица најголемиот член е? -прв излегува.
9. Комплексноста на пристапувањето до елементи преку содржина во низа? - $O(n)$.

10. На едно CD има N песни секоја со меморија SI , cd е преполно дали за бришење на песни за да се ослободи меморија е точно да се користи greedy? - Не.

11. Што е точно за дрво? - До секој јазол има различен пат почнувајќи од коренот.

12. Колкава е сложеноста за хеш табела со затворени кофички? - $O(1)$ а ако се понудено и $O(1)$ и $O(n)$ и двете ќе си ги заокружиш.

13. На кој принцип работи стекот? - LIFO

14. Кога е најнеповолно да користиме матрица на соседство и дадени број на јазли и ребра. Најнеповолно е да користите матрица на соседство кога имате голем граф со многу јазли и ребра а поготово кога графот е разрсен (sparse)

15. Што е точно за В-дрвата? - Кај В-дрвата сите термални јазли се на исто ниво.

16. Комплексноста да стигнеш до елемент во неподредена низа? - $O(n)$.

17. Д
-пребарување во ширина
-внатрешен јазол
-има најмалце едно дете.

18. Што функција има ОВНТ хеш со повеќе од 1 step? - помалку кластери.

19. Комплексност за бришење на првиот елемент од листата? - $O(1)$.

20. Комплексност за додавање елемент во низа? - $O(n)$.

21. При еднакратна ротација на јазол? - Inorder изминување не се менува.

22. Матрица на соседство е најповолна ако имаме? - најмал број на јазли и што поголем број на ребра.

23. Имплементација на стек со помош на листа? - Првиот елемент на листата е врв на стекот и стекот е неограничен при имплементација со листата.

2. Složenost na pristapuvanje na element spored indeks vo niza?

- $O(1)$

3. Koga e najdobro da se koristi dinamičko programiranje? -памятење

4. So sto najdobro bi se prevrtile elementi?

- Stek

5. Za adresar preziminja sto se koristi?

- B drvo

6. Koga graf e prost?

- Koga nema ciklusi i paralelni rebra

7. Dali ako nema mesto vo CD togas so alcen bi se dobile optimalni rezultati?

- Ne

8. Složenost na algoritmi ne se meri so?

- Nitu eden od ponudenite odgovori a bea dadeni brzina, kapacitet i uste nesto so broj..

9. Kaj prioriteta redica najgolemiot clen:

a) prv vleguva

b) prv izleguva ----- Tocen

c) ne izleguva

d) prv vleguva i posleden izleguva.

10. Sto e prost graf ?

- Graf vo koj so nema jamki i paralelni rebra.

11. Kompleksnost na pristapuvanje do element preku soдржина vo niza?

- $O(n)$

12. Na edno cd ima N pesni sekoja so memorija S_i , cdto e prepolno dali za brisenjeto na pesni za da se oslobodi memorija e točno da se koristi greedy ?

- Ne

13. Sto e točno od slednite za drvo ?

- Do sekoj jazol ima razlicen pat pocnuvajki od koren.

14. Kolkava e složenosta za hesh tabela so zatvoreni koficki?

$O(1)$ i $O(n)$

15. Zasto služi ciklična niza?

- Kaj redica se koristi za nadziranje na kompleksnosti.

16. Koga grafot e prost?

-Nema jamki i paralelni rebra.

17. Daden e netezinski i neorientiran graf. Algoritam za minimalno rastojanje od jazol s do bilo koj drug jazol

- Prebarivanje po sirini

18. Vnatresen jazol na drvo

- Ima najmalce jedno dete.

19. Kompleksnost da stignes do element za element vo nepodredena niza?

- $O(n)$

20. Shto e točno za b drvata?

- Kaj b drvata site terminalni jazli se na isto nivo.

21. Na koj princip raboti stekot?

- LIFO (Last-In-First-Out)

22. Koga e najnepovolno da koristime matrica na sosedstvo?

-Vo odgovorot dadeni se broj na jazli i rebra (treba brojot na jazli da e shto pomal a brojot na rebra pogolem).

24. Sto funkcija OBHT hash so povekje od 1 Step?

- Pomalku klasteri.

25. Koj e minimalniot broj na jazli kaj binarno drvo so visina d?

- $d+1$

26. Shto e točno kaj grafot:

- Teme koe sto ne pripaga na nikoj par rebra (u,v) e izolirano teme

27. Kompleksnost za dodavanje element vo niza :

- $O(n)$

28. Pri ednokratna rotacija na jazel :

- Inoreder izminuvanje ne se menuva (a imase ponudeni, Inoreder izminuvanje se menuva, preorder izminuvanje se menuva, preorder izminuvanje ne se menuva)

29. Implementacija na stek so pomos na lista:

- Prvot element na listat e vrv na stekot i stekot e neogranicen pri implementacija so lista.

30. Matrica na sosedstvo e najpovolna ako imame:

- Najmal broj na jazli i sto pogolem broj na rebra

31. Greedy ne se koristi...Slicno bese prasanjKolkava e kompleksnosta na best fit reshenieto kaj knapsack algoritmot ? $O(n^2)$ eto so toa od kolkvium za dijagnosticiranje na bolesti, ama so mereneje na temeperatura na zemjata bese

32. Koja e slozenosta za pristap do element vo nepodredena niza?

- $O(n)$

33. Koga principot na razdeli i vladej stanuva neefikasen?

34. Brojot na neterminalni jazli kaj drvo so stepen 3 e pomal od brojot na terminalni jazli.

37. Asimptotska gorna granica na funkcijata $f(n)=8n+1$ NE e funckijata?

a) $g(n)=n*n$;

b) $g(n)=n*n*n$;

c) $g(n)=8n+2$;

d)Nitu edno od ponudenite točno

38. Kolkava e kompleksnosta za brishenje na element od dvostrana lista ako prethodno e pokazan elementot? $O(1)$

39. Kolkava e kompleksnosta dokolku sakate da izbrisete element od krajot na nizata?

- $O(1)$



Најдобар случај – бришење елемент на крај од низа

- Нема потреба од извршување на чекор 1
- Се извршува само чекор 2, една операција

Сложеноста на алгоритмот е $O(1)$.

40. Koga se koristi dinamicko programiranje?

- Koga problemite se preklopuvaat

41. Sto e redica (taka nesto)?

- Na krajot se dodava element od pocetokot se vadi.

42. Shto e niza?

43. Kompleksnost da se izbrishi prviot element vo lista?

- $O(1)$

44. Vo edno CD so m megabajti treba da se zapishat pesni. Vкупnite MB na pesnite se pogolemi od m. Dali Greedy algoritamot e najefikasen vo ovoj sluchaj?

- Ne

45. Idejata na INSERTION SORT:

- Vo prethodno vekje sortirana podniza

46. Kompleksnosta na algoritmot od shto NE zavisi?

47. Kolkava e kompleksnosta na brishenje 1 element od lista?

- $O(n)$

48. Koj od ovie ima najgolema kompleksnost?

- (I tuka opcii od $O(n)$, $O(n^2)$ i sl) - $O(n^2)$ најголема комплексност

49. Koga e dobro razdeli pa vladej?

50. Koj algoritam e najdobar za prevrtuvanje na niza?

-stek?

51. Od datoteka se sortirat 200 iminja za 200ns so bubble sort...kolku vreme treba za da 800 iminja?

- 3 200ns

52. Prevrtuvanje na elementi vo niza e prinzip slicen na...

- Stek

53. Koga stanuva razdeli i vladej ne efikasen?

- Koga podproblemite se preklopuvaat

54. Koga se upotrebuva dinamičko programiranje?

- Koga problemite se preklopuvaat

55. Golemo omega e :

- Dolna granica (...izvršuvanje na programata vo najdobar slučaj)

56. Ako treba da se dijagnosticira nekoja bolest , dali treba da se upotrebi alčen algoritam?

-Ne

57. Koga se upotrebuva bubble sort?

брзо и едноставно сортирање на мали податочни множества.

58. Isti best case and worst case kaj koi sortiranja:

- Merge и maximumEntry mislam bea +Counting, Heap, Radix

59. Koj od slednite e najdobro resenje za prevrtuvanje na redosled na elementi?

- Stek

60. Koj od ovie algoritmi najsporo ke se realizira?

- $O(2^n)$

$O(n \cdot n \cdot n)$

i uste dve opcii ne pamtam..ama odgovorot bese pod A.... (2^n)

61. Sortiranje na niza od koi 4 iteracii bile vekje izvršeni daden primer 1 2 4 5 3 8 7), da se odbere so koj algoritam se sortira:

- (Insertion).

1. Колкава е комплексноста доколку сакате да избришете елемент од крајот на низата --- $O(1)$

2. Колкава е комплексноста доколку сакате да избришете првиот елемент од листа --- $O(1)$

3. Кој од понудените има најголема комплексност? (понудени се: $O(n)$, $O(n^2)$, $O(2^n)$,)
4. Асимптотска горна граница на функцијата $f(n)=8n+1$ НЕ е функцијата?
- a) $g(n)=n*n$;
 - b) $g(n)=n*n*n$;
 - c) $g(n)=8n+2$;
 - d) Ниту едно од понудените
5. Кој од овие алгоритми најспоро ќе се реализира?
- a) $O(2^n)$
 - b) $O(n*n*n)$
6. Што е низа? --- последователно множество на мемориски локации, т.е. множество на подредени парови
- (индекс, вредност), при што за секое појавување на индекс, постои соодветна вредност асоцирана за тој индекс
7. Што е ред? --- последователно множество на мемориски локации множество на подредени парови (индекс, вредност) при што за секое појавување на индекс, постои соодветна вредност асоцирана за тој индекс
8. Кога се користи динамичко програмирање? --- кога проблемите се преклопуваат
9. Во едно CD со m megabajti треба да се запишат песни. Вкупните MB на песните се поголеми од m . Дали
10. Greedy алгоритмот е најефикасен во кој случај?
11. Идејата на INSERTION SORT е користење во: --- скоро сортирана низа
12. Комплексноста на алгоритмот од што НЕ зависи?
13. Колкава е комплексноста на Best Fit решението кај Knapsack алгоритмот ? – $O(n^2)$, но може и $O(n \log n)$

14. Kolkava e kompleksnosta za brishenje na element od dvostrana lista ako prethodno e pokazan elementot $O(1)$

15. Кога е најдобра да се примени алгоритмот раздели па владеј? – кога проблемот може да се подели на подпроблеми, а притоа тие да се дисјунктни.

16. Кој алгоритам е најдобар при превртување на низа?

17. Од датотека се сортираат 200 имиња за 200ns со Bubble Sort. Колку време треба за да се сортираат 800 ? --- 3200ns (За тоа што ни е дадено, имаме дека $n=200$. Односно, кажано ни е дека $O(200*200)=200ns$.

За n^2 имаме дека $n^2=800$. $800/200=4$, што значи дека факторот на зголемување на input-от ни е 4, па имаме:

$$O(n^2*n^2)=O(800*800)=O(4*200*4*200)=O(16*200*200) = 16*O(n*n)=16*200=3200ns)$$

18. Превртување на елементи во низа е принцип сличен на ... --- стек

19. Кога алгоритмот Раздели и владеј станува НЕ ефикасен?

кога ќе се примени директно на проблем чии потпроблеми не се независни како на пример горниот пример со Фибоначиевите броеви добиваме неефикасен алгоритам во кој се повторуваат беспотребни пресметувања

20. Golemo omega е : асимптотска долна граница(најдобро извршување) на некоја $f-j$

21. Ако треба да се дијагностицира некоја болест, дали треба да се употреби алчен алгоритам? --- Не

22. Кога се употребува Bubble Sort?

ako vekje e skoro sortirana(skoro-sortirana niza $O(n)$ ili vekje sortirana nizata?

23. Исти best case and worst case кај кои сортирања постојат:
- merge и maximum Entry „heap,selection

24. факонудените е најдобро решение за превртување на редослед на елементи --- стек

25. Сортирање на низа од кој 4 итерации биле веќе извршени (даден пример 1 2 4 5 3 8 7), да се одбере со кој алгоритам се сортира (Insertion).

Za adresar preziminja sto se koristi -B drvo

dali ako nema mesto vo CD togas so alcen bi se dobile optimalni rezultati? Ne

Slozenost na algoritmi ne se meri so? Nitu eden od ponudenite odgovori a bea dadeni brzina, kapacitet i uste nesto so broj..

kaj prioritetna redica najgolemiot clen:

a)prv vleguva

b)prv izleguva ---- Tocen

c)ne izleguva

d)prv vleguva i posleden izleguva.

Комплексност за бришење првиот елемент од листа:

- $O(1)$

Што функција има ОБНТ hash со повеќе од 1 Step

- Помалку кластери

Koj e minimalniot broj na jazli kaj binarno drvo so visina d? - $d+1$

Shto e tochno kaj grafot: teme koe sto ne pripaga na nikoj par rebra (u,v) e izolirano teme

1.Kompleksnost za dodavanje element vo niza : $O(n)$

2. Pri ednokratna rotacija na jazel :Inoreder izminuvanjeto ne se menuva

3. Implementacija na stek so pomos na lista: prviot element na listat e vrv na stekot i stekot e neogranicen pri implementacija so lista

5. Greedy ne se koristi...Slicno bese prasanjeto so toa od kolkvium za dijagnosticiranje na bolesti, ama so mereneje na temeperatura na zemjata bese

1. koja e slozenosta za pristap do element vo nepodredena niza $O(n)$

2. dadena matrica [1, 2, 1, 0, 1, 2, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0] vo hash tabelata treba da e 1) 1, 2, 2, 3 vaka neso bese

1. Kolkava e kompleksnosta na best fit reshenieto kaj knapsack algoritmot ?

- $O(n^2)$

2. Asimptotska gorna granica na funkcijata $f(n)=8n+1$ NE e funckijata?

a) $g(n)=n*n$;

b) $g(n)=n*n*n$;

c) $g(n)=8n+2$;

d)Nitu edno od ponudenite

3. Kolkava e kompleksnosta za brishenje na element od dvostrana lista ako prethodno e pokazan elementot

- $O(1)$

Hash kompleksnost na prebaruvanje/dodaвање/бришење во Просечен случај:

- $o(1)$

Алчно пристапување за мерење на температура на ракета:

- Не

***Vo stekot elementite se pristapuvaat:**

zavisno od implementacija

od vrvot -- točno

od krajot

zavisno od golemina

***Algoritam za otkrivanje na ciklus vo graf:**

DFS - dfs e za otkrivanje na ciklus

BFS - bfs za shortest path

Dijkstra

minimal spanning tree

***best effort i worst effort isto e kaj : (tuka bea site tie bubble sort, insertion sort ... itn)**

sto e prost graf ?

- graf vo koj so nema jamki i pararelni rebra

kompleksnost na pristapuvanje do element preku soдрzina vo niza?

- $O(n)$

Na edno cd ima N pesni sekoja so memorija S_i , cdto e prepolno dali za brisenjeto na pesni za da se oslobodi memorija e točno da se koristi greedy ?

-ne

Sto e točno od slednite za drvo ?

-Do sekoj jazol ima razlicen pat pocnuvajki od korenот.

kolkava e slozenosta za hesh tabela so zatvoreni koficki?

$O(1)$ i $O(h)$

Koga grafot e prost?

-Nema jamki i paralelni rebra

- **Daden e netezinski i neorientiran graf. Algoritam za minimalno rastojanie od jazol s do bilo koj drug jazol**

-prebaruvanje po sirina

- **Vnatresen jazol na drvo**

-ima najmalce edno dete.

- **Kompleksnosta da stignes do element za element vo nepodredena niza?**

- $O(n)$

- **Shto e točno za b drvata?**

-kaj b drvata site terminalni jazli se na isto nivo.

- **Комплексност за бришење првиот елемент од листа:Што функција има O(1) hash со повеќе од 1 Step**

- Помалку кластери

- $O(1)$

Koj e minimalni broj na jazli kaj binarno drvo so visina d? - d+1

Shto e točno kaj grafot: teme koe sto ne pripaga na nikoj par rebra (u,v) e izolirano teme

1. Kompleksnost za dodavanje element vo niza : $O(n)$

Implementacija na stek so pomos na lista: prviot element na listat e vrv na stekot i stekot e neogranicen pri implementacija so lista

Matrica na sosedstvo e najpovolna ako imame: najmal broj na jazli i sto pogolem broj na rebra

5. Greedy ne se koristi...Slicno bese prasanjeto so toa od kolkvium za dijagnosticiranje na bolesti, ama so mereneje na temeperatura na zemjata bese

1. koja e slozenosta za pristap do element vo nepodredena niza

- $O(n)$ mislam / $O(1)$ e slozenosta za pristap do element vo niza bez razlika dali e podredena ili nepodredena

2. dadena matrica [1, 2, 1, 0, 1, 2, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0] vo hash tabelata treba da e 1) 1, 2, 2, 3 vaka neso bese

4. brojot na neterminalni jazli kaj drvo so stepen 3 e pomal od brojot na terminalni jazli

1. Kolkava e kompleksnosta na best fit reshenieto kaj knapsack algoritmot ?

- $O(n^2)$

2. Asimptotska gorna granica na funkcijata $f(n)=8n+1$ NE e funkcijata?

a) $g(n)=n*n$;

b) $g(n) = n * n * n$;

c) $g(n) = 8n + 2$;

d) Site: od ponudenite <-

3. Kolkava e kompleksnosta za brishenje na element od dvostrana lista ako prethodno e pokazan elementot

$O(1)$

1. Koja e slozhenosta na

```
int power(int x, int n){  
    if(n == 0) return 1;  
    if(n == 1) return x;  
    return x * power(x,n-1); }
```

Одговор : $O(N)$

2. Со која сложеност се наоѓа член во не-сортирана низа

Одговор : $O(N)$

3. Ојлеров пат е :

Одговор: Пат кој го користи секое ребро од графот точно еднаш

4.Што важи В-дрва

Одговор: Сите листови се на исто ниво

5. Во minhear имате 3 члена, кое од следниве е точно:

- Се печати сортирана низа во inorder
- Се печати сортирана низа во preorder
- Се печати сортирана низа во postorder
- Ниту едно од наведените (ова е точниот одговор)

1.

int suma = 0;	1	
for(int i = 1; i <= n; ++i)	$1 + n + 1 + n$	
suma = suma*i + i	$n * (1 + 1 + 1)$	
return suma	1	$\Rightarrow 5n + 4 = O(n)$ (neli e vaka? 🤔)

Koj e brojot na operaciji i kompleksnosta na algoritamot?

- $f(n) = 5n + 4$, $O(n)$



2.

Koj od slednite iskazi e točno za drva?

-Visinata e najdolgata pateka od korenот до listovite.

3. Sto e točno za prioriteta redica pretstavena so nepodredena niza?

-Dodavanje na element $O(1)$, vadenje na element $O(n)$.

(ostanate bea kombinacii od prethodnite)

4. Koi od ponudenite se modeliraat so grafovi?

- Proteinski vrski

- Sahovska tabla

- (ne mozam da se setam na ovoj ponuden odgovor)

- Site prethodni <- Točno

5. Koi parametri se bitni kaj algoritmite za sortiranje?

-Brzina i memorija

1. Sto e redica?

- Редицата преставува еднодимензионална линеарна секвенца од елементи која работи по принципот прв-внесен-прв-изваден.

2. Slozenost na pristapuvanje na element spored indeks vo niza?

- $O(1)$

3. Koga e najdobro da se koristi dinamičko programiranje?

-Кога има проблеми што се преклопуваат

4. So sto najdobro bi se prevrtile elementi?

- Stek

5. Za adresar preziminja sto se koristi?

- B drvo

6. Koga graf e prost?

- Koga nema ciklusi i paralelni rebra

7. Dali ako nema mesto vo CD togas so alcen bi se dobile optimalni rezultati?

- Ne

8. Slozenost na algoritmi ne se meri so?

**- Nitu eden od ponudenite odgovori a bea dadeni brzina, kapacitet i uste nesto so broj..
se meri so broj na operacii**

9. Kaj prioritetna redica najgolemiot clen:

a)prv vleguva

b)prv izleguva ----- Tocen

c)ne izleguva

d)prv vleguva i posleden izleguva.

10. Sto e prost graf ?

- Graf vo koj so nema jamki i pararelni rebra.

11. Kompleksnost na pristapuvanje do element preku soдрzina vo niza?

- $O(n)$ 👍

12. Na edno cd ima N pesni sekoja so memorija S_i , cdto e prepolno dali za brisenjeto na pesni za da se oslobodi memorija e točno da se koristi greedy ?

-Ne

13. Sto e točno od slednite za drvo ?

-Do sekoj jazol ima razlicen pat pocnuvajki od koren.

14. Kolkava e slozenosta za hesh tabela so zatvoreni koficki?

$O(1)$ i $O(h)$

15. Zosto služi ciklicna niza?

- Kaj red se koristi za nadminuvanje na kompleksnosta.

16. Koga grafot e prost?

-Nema jamki i paralelni rebra.

17. Daden e netezinski i neorientiran graf. Algoritam za minimalno rastojanje od jazol s do bilo koj drug jazol

- Prebaruvanje po sirina (BFS)

18. Vnatresen jazol na drvo

- Ima najmalce edno dete.

19. Kompleksnosta da stignes do element za element vo nepodredena niza?

- $O(n)$ 👍

20. Shto e točno za b drvata?

- Kaj b drvata site terminalni jazli se na isto nivo.

20.1 Sto e terminalen jazol ? (itno)

- Jazol sho nema deca(levo ili desno poddrvo) fala 😊👍

21. Na koj princip raboti stekot?

-Last-in-First-Out (LIFO)

22. Koga e najnepovolno da koristime matrica na sosedstvo?

-Vo odgovorot dadeni se broj na jazli i rebra (treba brojot na jazli da e shto pomal a brojot na rebra pogolem).

-Najnepovolno e koga imame golem broj na jazli, mal broj na rebra, a najpovolno e koga imame mal broj jazli, golem broj na rebra.

23. Комплексност за бришење првиот елемент од листа:

- $O(1)$ 👍

24. Sto funkcija OBHT hash so povekje od 1 Step?

- Pomalku klasteri. 👍

25. Koj e minimalniot broj na jazli kaj binarno drvo so visina d?

- $d+1$ 👍

26. Shto e točno kaj grafot:

- Teme koe sto ne pripaga na nikoj par rebra (u,v) e izolirano teme

27. Kompleksnost za dodavanje element vo niza :

- $O(n)$ 👍

28. Pri ednokratna rotacija na jazel :

- Inoreder izminuvanjeto ne se menuva (a imase ponudeni, Inoreder izminuvanjeto se menuva, preorder izminuvanjeteo se menuva, preorder izminuvanjeto ne se menuva)

29. Implemetacija na stek so pomos na lista:

- Prvot element na listat e vrv na stekot i stekot e neogranicen pri implementacija so lista.

30. Matrica na sosedstvo e najpovolna ako imame:

- Najmal broj na jazli i sto pogolem broj na rebra 👍

31. Greedy ne se koristi...Slicno bese prasanjeto so toa od kolkvium za dijagnosticiranje na bolesti, ama so mereneje na temeperatura na zemjata bese

32. Koja e slozenosta za pristap do element vo nepodredena niza?

-/ $O(n)$

34. Brojot na neterminalni jazli kaj drvo so stepen 3 e pomal od brojot na terminalni jazli.

- kako e ova razbiras ??
- neznam
- to e

35. Sto funkcija ima so poveke od 1 step (dvojno heshiranje)?

- sto pomalku klasteri

36. Kolkava e kompleksnosta na best fit reshenieto kaj knapsack algoritmot ?

$O(n^2)$

37. Asimptotska gorna granica na funkcijata $f(n)=8n+1$ NE e funckijata?

a) $g(n)=n*n$;

b) $g(n)=n*n*n$;

c) $g(n)=8n+2$;

d) Nitu edno od ponudenite ←

38. Kolkava e kompleksnosta za brisenje na element od dvostrana lista ako prethodno e pokazan elementot?

- $O(1)$

39. Kolkava e kompleksnosta dokolku sakate da izbrisete element od krajot na nizata?

- $O(1)$ (ako e posleden)
- $O(n)$ (ako e ne-posleden)

40. Koga se koristi dinamicko programiranje?

- Koga problemite se preklopuvaat

41. Sto e red (taka nesto)?

- Na krajot se dodava element od pocetokot se vadi.
- Se dodava na opas se vadi od glava

42. Shto e niza?

- niza e niza so ciklus nesto i vrtime ?? (-2)
- последователно множество на мемориски локации (+2)

43. Kompleksnost da se izbrishi prvot element vo lista?

- $O(1)$

44. Vo edno CD so m megabajti treba da se zapishat pesni. Vкупните MB na pesnite se pogolemi od m. Dali Greedy algoritmot e najefikasen vo ovoj sluchaj?

- Ne, Da ?????

45. Idejata na INSERTION SORT:

- Vo prethodno vekje sortirana podniza

46. Kompleksnosta na algoritmot od shto NE zavisi?

- зависи od број на операции

47. Kolkava e kompleksnosta na brisenje 1 element od lista?

- $O(n)$

48. Koj od ovie ima najgolema kompleksnost?

- (I tuka opcii od $O(n)$, $O(n^2)$ i sl)

49. Koga e dobro razdeli pa vladej?

Divide and Conquer should be used when same subproblems are not evaluated many times. Otherwise Dynamic Programming or Memoization should be used. For example, Binary Search is a Divide and Conquer algorithm, we never evaluate the same subproblems again.

50. Koj algoritam e najdobar za prevrtovanje na niza?

- stek

51. Od datoteka se sortirat 200 iminja za 200ns so bubble sort...kolku vreme treba za da 800 iminja?

- 3200ns (moze objasnenie?)

52. Prevrtovanje na elementi vo niza e princip slicen na...

- Stek

53. Koga stanuva razdeli i vladej neefikasen?

- Koga podproblemite se preklopuvaat

54. Koga se upotrebuva dinamicko programiranje?

- Koga problemite se preklopuvaat

55. Golemo omega e : Dolna granica (...izvrsuvanje na programata vo najdobar slucaj)

56. Ako treba da se dijagnosticira nekoja bolest , dali treba da se upotrebi alcen algoritam?

-Ne

57. Koga se upotrebuva bubble sort?

It's useful for smaller sets of elements but is inefficient for larger sets. As far as being the fastest on an extremely small and/or nearly sorted set of data, while that can cover up the weakness of bubble sort (to at least some degree), an insertion sort will essentially always be better for either/both of those.

58. Isti best case and worst case kaj koi sortiranja:

- Merge и maximumEntry mislam bea

59. Koj od slednite e najdobro resenije za prevrtuvanje na redosled na elementi?

- Stek

60. Koj od ovie algoritmi najsporo ke se realizira?

- $O(2^n)$

$O(n \cdot n \cdot n)$

i uste dve opcii ne pamtam..ama odgovorot bese pod A.... (2^n)

61. Sortiranje na niza od koi 4 ite+4φracii bile vekje izvrsheni daden primer 1 2 4 5 3 8 7), da se odbere so koj algoritam se sortira: Insertion lista

потоа, insertion sort е најдобро да се користи

- кога низата е скоро сортирана

наредно, која е сложеноста за вадење на прв елемент од листа

- $O(1)$

на knapsack проблем која му е сложеноста со bestfit

мислам $O(n^2)$ беше

Кај приоритетна редица, кое од следните тврдења е точно ако е подредена низа?

- додај $O(n)$, извади $O(1)$ <----

- додај $O(n)$, извади $O(n)$

- додај $O(1)$, извади $O(1)$

- додај $O(1)$, извади $O(n)$

Merge Sort se koristi koga:

-ni se potrebni zagarantirani performansi(ne mi teknuvaat drgi ponudeni,ova e tocnoto)

naogjanje na posleden element vo lista ima kompleksnost:

$O(n)$

ako sakame da iskoristime vo nasata aplikacija abstrakten podatocen tip treba:

treba da gi specificirame mnozestvoto vrednosti i operaciite nad tie vrednosti

Kaj Sortiranje kompleksnosta na algoritмотo (ili taka neso bese) zavisit od ? -brojot na izvrсени ednostavni operacii

Za golemo Omega notacija vazi: (- mislam deka bese odgovor definira dolna granica i nz koj bese vtoriot del od odgovorot xD)

1. Koja od ovie e asimptotska granica na funkciјата $f(x) = x^2 + 3x + 4$?

a. x

b. $x \cdot \log(x)$

c. $2x^3$

d. никое од понудените

2. Редица претставува:

Линеарна податочна структура каде елементите се додаваат од едната страна (опашот) а се пристапуваат од другата страна (главата).

3. Динамичкото програмирање се користи кога:

a. Повеќето проблеми се поклопуваат <---

b. Потпроблемите не се поклопуваат

2. Ако имаме n песни со вкупна големина од D и имаме цд со големина од M , за $D > M$, дали со алчен алгоритам можеме да определиме на цд-то да ставиме најголем број на песни?

Одговор: Да

1. Koja e kompleksnosta na kodot? *insert рекурзивна функција за пресметување степен на број here* - $O(n)$

2. Koј алгоритам за сортирање има исти best и worst-case комплексност?

-Merge sort и Maximum Entry sort.

3 . Стекот имплементиран со низа:

Неколку можни одговори

- Има ограничена должина која зависи од должин

ата на низата.

Koja od naveдените e горна граница на $f(n) = 8n + 1$?

a) $n^2 \cdot n$ <---

b) $\log n$

c) neshto drugo so pomala kompleksnost od $O(n)$ kako pod b

d) nitu edno od navedenite

Ако QuickSort сортира 1000 елементи за 100ns, за колку време отприлика би сортирал 100 елементи?

- 6.7ns <----

- 11.1ns

- 15.7ns

и уште три понудени неточни

****QuickSort има комплексност $O(n \log(n))$, односно за да сортира n елементи, потребни му се $n \log(n)$ чекори.

EEEE сеГаААА, знаеме дека за $n=1000$, алгоритмот се извршува за 100ns, односно $1000 \log(1000)$ инструкции = 100ns.

Станува збор за логаритам со основа 2, така да $\log(1000) = 9.966$ приближно, и од тука добиваме $1000 * 9.966$ инструкции = 100ns.

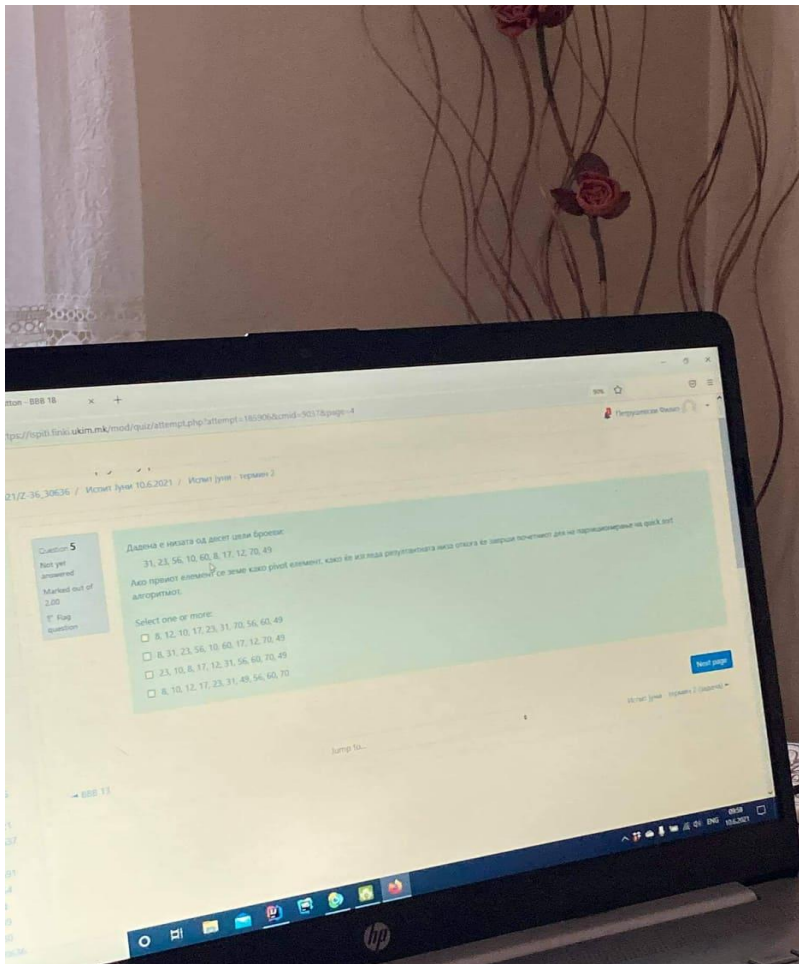
Од тука пресметуваме дека една инструкција се пресметува за приближно 0.01 ns.

Откако пресметавме колку време е потребно за една инструкција, можеме да го пресметаме времето за QuickSort со $n=100$, бидејќи овој алгоритам ќе има $100 \log(100)$ инструкции.

$\log(100)$ е 6.643 заокружено, така да за овој алгоритам ќе се потребни $6.643 * 100$ инструкции односно 664.3.

Ако го помножаме овој број со времето по инструкција кое го пресметавме погоре, се добива $664.3 * 0.01 \text{ ns} = 6.643 \text{ ns}$.

Ваљда бидејќи заокружував на три децимали, малку е различно моево решение од нивното, ама таа е постапката



pod C e (provereno od profesor)

предмети се претставени како подредени парови {тежина, вредност}: $\{\{10,60\},\{7,28\},\{4,20\},\{2,24\}\}$. Капацитетот на ранецот е 7

7. Да се најде максималната вредност препоставувајќи дека има само еден од секој предмет и предметите се делливи (fractional knapsack).

Select one:

a. 48

b. 44

c. 28

d. 54 <----- (се повторуваат прашањата со ранецот, ова е 54 и доста со истото прашање)

Ознака | Тежина | Вредност | Вредност/Тежина

i1	10	60	$60/10=6$
i2	7	28	$28/7=4$
i3	4	20	$20/4=5$
i4	2	24	$24/2=12$

Почнуваш од тој со најголем вредност/тежина.

=====Старт=====

Капацитет = 7

ВкупенПрофит = 0

Бираме i

$$\text{Капацитет} = 7 - 2 = 5$$

$$\text{ВкупенПрофит} = 24$$

Бираме i1

$$\text{Капацитет} = 5 - 5 = 0$$

$$\text{Профит} = (\text{ПреостанатКапацитет} / \text{ТежинаНа}i1) * \text{Вредност}$$

$$\text{Профит} = 5/10 * 60 = 0.5 * 60 = 30$$

$$\text{ВкупенПрофит} = 24 + 30 = 54.$$