СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
HS Algorithm 2022

шифра

Број бодова по задацима

1.		6.	
2.		7.	
3.		8.	
4.		9.	
5.		10.	

Укупан број поена:

Задатак 1. - Линеарна алгебра

Линеарна алгебра је математичка област која се бави изучавањем линеарних појава у математици. Раније сте се већ сусрели са централним појмом у линеарној алгебри - вектором и то најчешће са његовом геометријском интерпретацијом као усмерене дужи која има свој почетак и крај (самим тим правац и смер) и интензитет (тј. дужину). Међутим, постоји и алгебарска интерпретација вектора, а најлакше ћемо ту интерпретацију објаснити увођењем појма векторског простора (над пољем реалних бројева):

Дефиниција. Нека је V непразан скуп и нека је дефинисано пресликавање $+\colon V \times V \to V$ (ову операцију називамо сабирањем вектора) као и пресликавање $\cdot\colon \mathbb{R}\times V \to V$ (ово дејство називамо множење вектора скаларом) тако да за важи:

1. За све $v, u \in V$ је $v + u \in V$ - збир два вектора је вектор.
2. За све $v, u, w \in V$ је $(v + u) + w = v + (u + w)$ - сабирање вектора је асоцијативно
3. Постоји $e \in V$ тако да за свако $v \in V$ важи $v+e = e+v = v$ - постоји нула вектор (чијим додавањем се ништа не мења)
4. За свако $v \in V$ постоји $-v \in V$ тако да $v + (-v) = (-v) + v = e$ - за сваки вектор постоји његов супротан вектор
5. За све $u, v \in V$ је $u + v = v + u$ - редослед сабирања вектора није битан
6. За све $\lambda \in \mathbb{R}$ и све $v, u \in V$ је $\lambda \cdot (u + v) = \lambda \cdot c + \lambda \cdot v$ - дистрибутивност сабирања вектора према множењу вектора скаларом
7. За све $\lambda, \mu \in \mathbb{R}$ и све $v \in V$ је $(\lambda + \mu) \cdot v = \lambda \cdot v + \mu \cdot v$ - дистрибутивност сабирања вектора према множењу вектора скаларом
8. За све $\lambda, \mu \in \mathbb{R}$ и све $v \in V$ је $\lambda \cdot (\mu \cdot v) = (\lambda\mu) \cdot v$ - множење вектора скаларима је једнако множењу вектора производом скалара
9. За све $v \in V$ је $1 \cdot v = v$ - множење јединицом не мења вектор

Елементе скупа V називамо векторима док елементе скупа $\mathbb{R}$ називамо скаларима.

Посматрајмо сада скуп $V = \mathbb{R}^2$ и нека је $(x_1, x_2) + (y_1, y_2) = (x_1 + y_1, x_2 + y_2)$ као и $\lambda \cdot (x_1, x_2) = (\lambda x_1, \lambda x_2)$. Овако дефинисан V и операција сабирања као и дејство множења јесте један векторски простор. Зашто су нам вектори тј. векторски простори корисни? Врло лако можемо записивати одређене информације помоћу њих - на пример уколико скупљамо информације о становима и то о цени стана и његовој површини, те информације можемо записати као вектор у $\mathbb{R}^2$ где је прва компонента цена стана, а друга његова површина. Овде ћемо на примеру простора полинома највисе трећег степена идејно проучити особине векторских простора. Векторски простор полинома највише трећег степена је $\mathbb{R}_3[x] = \{p|p(x) = a_0 + a_1x + a_2x^2 + a_3x^3, a_i \in \mathbb{R}\}$ са дефинисаним операцијама сабирања вектора

$$(a_0 + a_1x + a_2x^2 + a_3x^3) + (b_0 + b_1x + b_2x^2 + b_3x^3) = (a_0 + b_0) + (a_1 + b_1)x + (a_2 + b_2)x^2 + (a_3 + b_3)x^3$$

и множење вектора скаларом

$$\lambda \cdot (a_0 + a_1x + a_2x^2 + a_3x^3) = \lambda a_0 + \lambda a_1x + \lambda a_2x^2 + \lambda a_3x^3$$

(a) Сваки векторски простор има своју базу. Шта је база? То је скуп линеарно независних вектора $\{e_1, e_2, \dots, e_n\}$ (то значи да се ниједан вектор из базе не може изразити у виду линеарне комбинације осталих вектора из те базе тј. не може се десити да $e_1 = \lambda_2 e_2 + \dots \lambda_n e_n$) такав да се сви вектори из датог векторског простора могу записати у виду $v = \lambda_1 e_1 + \lambda_2 e_2 + \dots + \lambda_n e_n$. У векторском простору полинома највише трећег степена једна база је $(1, x, x^2, x^3)$. База векторског простора није јединствено одређена, али знамо да су све базе истобројне и та величина зове се димензијом векторског простора. У нашем случају, димензија векторског простора полинома степена највише трећег степена је 4. Дати су скупови полинома:

$$E_1 = (1+x, x+x^2, x^2+x^3, 1+3x+3x^2+x^3), \quad E_2 = (x, 1+x^2, 3x+x^3, x^3),$$

$$E_3 = (x+x^2, 1+x^3, x^2-x, 1-x^3)$$

Међу скуповима E_1, E_2, E_3 бази векторског простора полинома степена највише 3 има? **[4 поена]**

а) 0; б) 1; в) 2; г) 3;

(б) Када одредимо базу e датог векторског простора, можемо доказати да је репрезентација неког вектора v преко вектора базе јединствена. Коефицијенти $\lambda_1, \lambda_2, \dots, \lambda_n$ који се налазе уз векторе из базе $e_1, e_2, \dots e_n$ називају се координатама вектора v у бази e и пишемо $[v]_e = (\lambda_1, \lambda_2, \dots, \lambda_n)$. Јасно $[v]_e + [u]_e = [v+u]_e$ као и $\lambda[v]_e = [\lambda \cdot v]_e$. У нашем примеру, координате вектора $1+x+3x^3$ у бази $(1, x, x^2, x^3)$ су $(1, 1, 0, 3)$. Дата је база векторског простора полинома највише трећег степена $e = (1, 1+x, 1+2x+x^2, 1+3x+3x^2+x^3)$. Ако су (a, b, c, d) координате вектора $1+2x+4x^2+8x^3$ у овој бази (коефицијент a одговара вектору 1, коефицијент b одговара вектору $1+x$ итд.), тада је вредност израза $a+2b+3c+4d$ једнака? (Помоћ: поставите систем линеарних једначина и решите га Гаусовом методом или одредите координате вектора $1, x, x^2, x^3$ па нађите координате датог вектора) **[5 поена]**

а) 0; б) 1; в) 2; г) 3;

Задатак 2. - Математичка анализа

Анализа је математичка област која се бави изучавањем граничних процеса и бесконачно малих величина. Вероватно сте чули за појмове као што су интеграл, извод или лимес (гранична вредност). Најбитнији појам у анализи јесте појам функције тј. пресликавања са којим сте се сусрели у средњешколском образовању. У анализи уобичајено посматрамо како се нека функција једне реалне промењиве понаша у "околини неке тачке a". Под "околином неке тачке a" мислимо на тачке које се налазе на неком (малом) растојању од тачке a (сетите се - математичка анализа изучава мале величине). На пример, интервал (1.95, 2.05) јесте околина тачке 2. Међутим, поменули смо појам растојања једне тачке од друге тачке - у скупу реалних бројева $\mathbb{R}$ је природно посматрати растојање две тачке тј. два броја a и b као апсолутну вредност њихове разлике $|a-b|$ - растојање бројева 2 и 2.05 је $|2-2.05| = 0.05$. У скупу $\mathbb{R}^2$ који можемо посматрати као раван, растојање меримо Питагорином теоремом - растојање тачака $A(x,y)$ и $B(z,t)$ је $\sqrt{(x-z)^2+(y-t)^2}$ - нацртајте слику!

Ова растојања између тачака скупа $\mathbb{R}$ и $\mathbb{R}^2$ су нам интуитивно јасна и поприлично позната из свакодневног живота. Како у математици увек тежимо општости тј. универзалности, често покушавамо да меримо и растојање између неке две тачке (тј. нека два елемента) произвољног скупа X. Посматрајмо особине растојања $\sqrt{(x-z)^2+(y-t)^2}$ између тачака $A(x,y)$ и $B(z,t)$ у $\mathbb{R}^2$: 1. растојање је ненегативно и једнако је 0 ако и само ако меримо растојање неке тачке од саме себе; 2. растојање од A до B је једнако растојању од B до A; 3. растојање од тачке A до B је мање него збир растојања тачака A и B од неке тачке $C(p,q)$ у $\mathbb{R}^2$. Сада можемо на произвољном скупу X одредити услове да нека функција $d: X \times X \to \mathbb{R}$ буде растојање тј. метрика (d је од енглеског distance)

Дефиниција. Нека је X произвољан (непразан) скуп. Функција $d\colon X \times X \to \mathbb{R}$ је **метрика на скупу** X ако задовољава следећа својства:

1. Позитивна дефинистност и недегенерисаност: За све $x, y \in X$ је $d(x,y) \geq 0$ и $d(x,y) = 0 \iff x = y$
2. Симетричност: За све $x, y \in X$ је $d(x,y) = d(y,x)$
3. Неједнакост троугла: За све $x, y, z \in X$ је $d(x,y) \leq d(x,z) + d(z,y)$

Тада кажемо да је (X,d) један **метрички простор**.

(а) Нека је $X = \mathbb{R}^3$. Као и у $\mathbb{R}^2$, метрику на $\mathbb{R}^3$ можемо дефинисати еуклидским растојањем:

$$d((x_1,x_2,x_3),(y_1,y_2,y_3)) = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+(x_3-y_3)^2}.$$

Међутим на истом скупу можемо дефинисати више метрика (мерења растојање не мора бити јединствено). Нека су дате функција $d_1, d_2, d_3\colon \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}$ дефинисане са:

$$d_1((x_1,x_2,x_3),(y_1,y_2,y_3)) = |x_1-y_1+x_2-y_2+x_3-y_3|$$

$$d_2((x_1,x_2,x_3),(y_1,y_2,y_3)) = |x_1-y_1|+|x_2-y_2|+|x_3-y_3|$$

$$d_3((x_1,x_2,x_3),(y_1,y_2,y_3)) = \log(1+(x_1-y_1)^2)$$

Од ових функција, метрика на скупу $\mathbb{R}^3$ има? **[3 поена]**

а) 0; б) 1; в) 2; г) 3;

(б) У метричком простору (X,d) за $a \in X$ и $r > 0$ дефинишемо скуп $B(a,r)$ који зовемо **отвореном куглом** са

$$B(a,r) = \{x \in X | d(x,a) < r\}$$

Овај појам нам помаже да дефинишемо појам отвореног тј. затвореног скупа у метричком простору X. Скуп $A \subseteq X$ је отворен ако за све тачке $a \in A$ постоји отворена кугла $B(a,r)$ (тј. постоји

$r > 0$) тако да $B(a, r) \subseteq A$. Скуп $A \subseteq X$ је затворен ако је $X \setminus A$ отворен. Ако је $X = \mathbb{R}^2$ где је метрика $d((x_1, x_2), (y_1, y_2)) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$ овај скуп је круг са центром у тачки $a(a_1, a_2)$ и полупречником r, без тачка са кружнице. У овом метричком простору су дати скупови $A_1 = \{(x, y) | y = 2x + 1\}$, $A_2 = \{(x, y) || x| + |y| < 1\}$, $A_3 = [0, 3] \times (-1, 1)$, $A_4 = B((0, 0), 3) \setminus \{(0, 0)\}$. Међу овим скуповима отворених у (X, d) има: **[3 поена]**

а) 0; б) 1; в) 2; г) 3;

док затворених има: **[3 поена]**

а) 0; б) 1; в) 2; г) 3;

Задатак 3. - Статистика

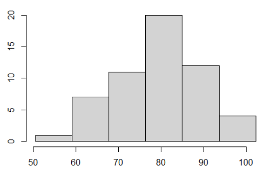
Статистика је наука о подацима. Бави се њиховим прикупљањем и анализом, презентовањем и закључивањем, као и доношењем одлука. Основни задатак статистичара је да предложи одговарајући математички модел којим би се подаци адекватно описали, након чега је могуће вршити даље анализе и предвиђања.

Популација је скуп јединки чије карактеристике изучавамо. Карактеристике које су предмет изучавања називамо *обележјима.* О њима најчешће сазнајемо на основу неког подскупа популације који називамо *узорак.* Уколико се подскуп бира насумично (сваки подскуп има неку вероватноћу да буде извучен) говоримо о *случајном узорку.* Поред случајности његова битна особина је *репрезентативност.* Потребно је да на основу њега можемо закључити о читавој популацији, као и да одабир чланова узорка не зависи од вредности обележја тих чланова. Ако је X посматрано обележје, са X_1, X_2,...,X_n ћемо означити случајан узорак.

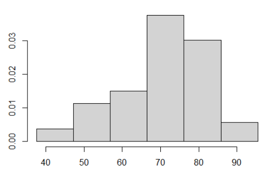
Пред вама се у наставку налази један експеримент и потребно је проћи кроз неке од основних корака у статистичкој анализи. Посматрајмо укупне поене на пријемном испиту за Математички факултет Универзитета у Београду 55 случајно изабраних ученика средњих школа у Србији. (Подаци су сортирани)

37.56	39.24	50.59	50.71	52.91	53.30	53.34	55.54	56.92	58.47	58.69
59.38	62.59	62.68	64.18	65.01	66.88	66.96	67.27	67.30	67.62	68.23
68.86	68.98	69.44	69.48	70.17	70.58	71.51	72.12	72.95	74.31	74.78
74.99	75.63	76.15	77.36	78.17	78.26	78.41	79.24	79.35	79.71	80.20
80.41	81.10	82.29	82.52	82.81	83.91	83.97	84.91	90.14	90.26	95.46

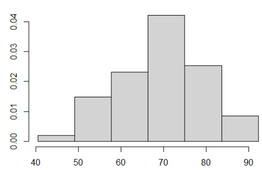
(а) У прелиминарној анализи помаже нам графички приказ обележја. За приказ нумеричког обележја користе се *хистограми.* За прављење хистограма потребно је да узорак групишемо у категорије (интервале), тако да сваки елемент узорка припада тачно једној категорији и одредимо број елемената из узорка који се налази у свакој од категорија. О броју и положају категорија одлучујемо ми као статистичари, а препорука је да имамо барем 5 категорија и да је број категорија $\lfloor \log_2 n \rfloor + 1$, где је n обим узорка. Величину категорија одређујемо на основу *распона* узорка $R = X_{(n)} - X_{(1)}$. Са $X_{(k)}$ се означава k-ти по реду елемент сортираног узорка. Затим, уколико имамо k категорија, њихова приближна величина је $\frac{R}{k}$. Хистограм управо представља графичи приказ учесталости по категоријама. Уколико су на y-оси фреквенције говоримо о *хистограму апсолутних фреквенција*, а уколико је приказан тај број подељен са величином узорка говоримо о *хистограму релативних фреквенција*, док уколико је то још подељено и са величином категорија, говоримо о *хистограму густине.* Хистограм густине за посматрани узорак укупног броја поена на пријемном испиту дат је са: **[3 поена]**



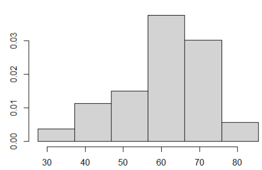
а)



б)



в)



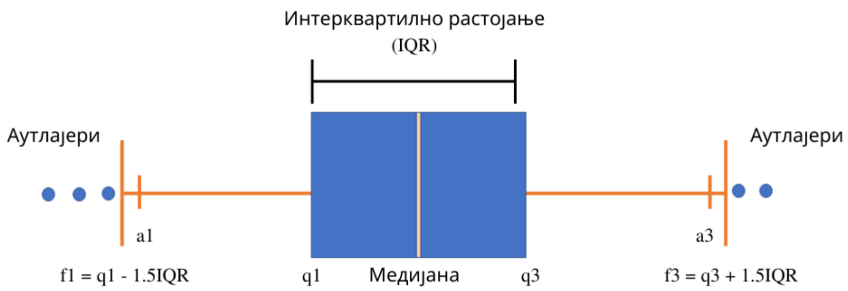
г)

(б) Следећи корак статистичке анализе је идентификација аутлајера. Под појмом *аутлајери* подразумевамо чланове узорка који се не уклапају у постојећи статистички модел. Представимо сада наше податке *кутијастим дијаграмом* (енг. boxplot).

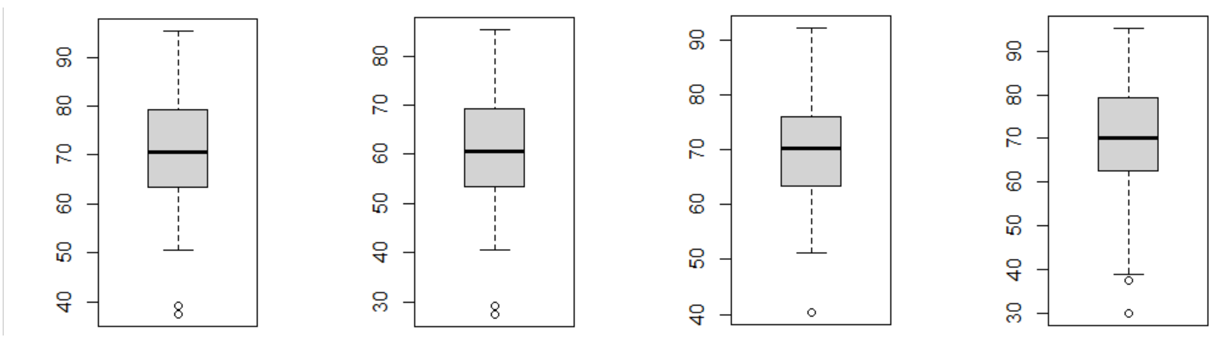
Уводимо ознаке:

- *Узорачка медијана* је $m_e = X_{(k+1)}$ ако је $n = 2k+1$ односно $m_e = \frac{X_{(k)}+X_{(k+1)}}{2}$ ако је $n = 2k$;
- q_1 и q_3 су *први* и *трећи квартил*. Ако је $n = 2k+1$, онда је q_1 узорачка медијана низа $X_{(1)},\ldots,X_{(k+1)}$, а q_3 узорачка медијана низа $X_{(k+1)},\ldots,X_{(2k+1)}$. Са друге стране, ако је $n = 2k$, онда је q_1 узорачка медијана низа $X_{(1)},\ldots,X_{(k)}$, а q_3 узорачка медијана низа $X_{(k+1)},\ldots,X_{(2k)}$;
- $\mathrm{IQR} = q_3 - q_1$ је *интерквартилно растојање*;
- $f_1 = q_1 - 1.5 \cdot \mathrm{IQR}$, $f_3 = q_3 + 1.5 \cdot \mathrm{IQR}$;
- $F_1 = q_1 - 3 \cdot \mathrm{IQR}$, $F_3 = q_3 + 3 \cdot \mathrm{IQR}$;
- a_1 је најмањи елемент узорка који је већи од f_1, a_3 је највећи елемент узорка који је мањи од f_3;

Елементи узорка који су између f_1 и F_1, односно f_3 и F_3 су благи аутлајери, док су они изван ових граница прави аутлајери.



Кутијасти дијаграм посматраног узорка за укупан број поена на пријемном испиту дат је на слици: **[3 поена]**



а) б) в) г)

(в) За крај овог статистичког истраживања упознаћемо се са тестирањем статистичких хипотеза о вредностима параметара. Основни елементи сваког статистичког теста су *нулта хипотеза* (H_0) и *алтернативна хипотеза* (H_1) (хипотеза која се прихвата уколико одбацујемо H_0). На нашем узорку поена са пријемног испита желимо да тестирамо хипотезу $\mathrm{H}_0 : m = 71$, где је m медијана коју има обележје X. Уводимо ознаку $T = \sum_{i=1}^{n} I\{X_i > 71\}$, при чему је I индикатор, те важи $I = 1$, уколико је испуњен услов $X_i > 71$, а иначе $I = 0$. Уколико је нулта хипотеза тачна број чланова узорка који су мањи од 71 треба да буде приближно једнак броју који су већи, односно, ако на основу узорка важи $|T - \frac{n}{2}| < 1$, онда закључујемо да је нулта хипотеза H_0 тачна. Који од следећих исказа је тачан: **[3 поена]**

а) Не одбацујемо нулту хипотезу $\mathrm{H}_0 : m = 71$.

б) Прихватамо алтернативну хипотезу $\mathrm{H}_1 : m \neq 71$.

в) Прихватамо алтернативну хипотезу $\mathrm{H}_1 : m \geq 71$.

г) Прихватамо алтернативну хипотезу $\mathrm{H}_1 : m \leq 71$.

Задатак 4. - Нумеричка математика и оптимизација

У свакодневном животу, када обрађујемо неке податке то радимо на неком дискретном скупу података тј. над коначно много података. Ове податке користимо на одређени начин да приближно одредимо шта се дешава ван тог дискретног скупа података - пример за то је интерполација функције.

Интерполација подразумева да полазну функцију $f(x)$ чије вредности знамо у коначном броју тачака $x_1, x_2, \ldots, x_n$ заменимо неком другом, интерполационом функцијом, чије се вредности поклапају са вредностима полазне функције на датом дискретном скупу тачака.

Кажемо да функција $g(x)$ интерполише функцију $f(x)$ ако је

$$g(x_k) = f(x_k),\ k = 0, \ldots n$$

Тачке $x_0, \ldots, x_n$ наизвају се чворови интерполације (и познате су нам вредности $f(x_k)$). Ако је функција $g(x)$ полином степена n, она се назива интерполациони полином степена n функције $f(x)$.

Lagrange-ов интерполациони полином степена n функције $f(x)$ одређен чворовима интерполације $x_0, \ldots, x_n$ за $x_i \neq x_j$, је

$$L_n(x) = \sum_{i=0}^{n} \left(\prod_{\substack{j=0 \\ j \neq 0i}}^{n} \frac{x - x_j}{x_i - x_j} \right) f(x_i) = \sum_{i=0}^{n} \left(\frac{x - x_0}{x_i - x_0} \cdot \frac{x - x_1}{x_i - x_1} \cdots \frac{x - x_{i-1}}{x_i - x_{i-1}} \cdot \frac{x - x_{i+1}}{x_i - x_{i+1}} \cdots \frac{x - x_n}{x_i - x_n} \right) f(x_i)$$

Ознака $\sum\limits_{i=0}^{n}$ представља суму $n+1$ елемената тј. $a_0 + a_1 + \ldots + a_n$, док $\prod\limits_{\substack{j=0 \\ j \neq i}}^{n} a_j$ представља производ свих a_j, за $j \neq i$, тј. $a_0 \cdot a_1 \ldots \cdot a_{i-1} \cdot a_{i+1} \cdot \ldots \cdot a_n$.

Сваки Lagrange-ов интерполациони полином је јединствен, то значи да уколико би постојао још један Lagrange-ов интерполациони полином $\mathcal{L}_n$ функције $f(x)$ важило би $L_n(x) = \mathcal{L}_n(x)$

(а) За фиксирано $n \in \mathbb{N}$ нека су $x_0, \ldots, x_n$ чворови интерполације. Дефинишимо функције $l_k \colon \mathbb{R} \to \mathbb{R}$ за $k = 0, \ldots n$ на следећи начин:

$$l_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^{n} \frac{x - x_j}{x_k - x_j} \psi(x_j) \text{ за } k = 0, \ldots, n \text{ где је } \psi(x) = \begin{cases} e^{-3}, & x_0 \leq x \leq x_n \\ 0, & \text{иначе} \end{cases}$$

Користећи дефиницију и особине Lagrange-овог полинома, колико износи $\sum\limits_{k=0}^{n} l_k(x)$? **[3 поена]**

а) 0; б) e^{-3}; в) e^{-3n}; г) 1;

(б) Уколико нам унапред није позната функција коју интерполирамо, већ само њене вредности у чворовима интерполације, и даље је могуће одредити 'приближну' вредност функције у некој произвољној тачки коришћењем Lagrange-овог полинома. Тако за неку вредност x, формирањем интерполационог полинома, одређујемо вредност у тој тачки као $L_n(x)$.

Међутим, често се дешава да имамо вредност функције $y = f(x)$ али нам је непозната вредност x. Такође и овде моземо применити интерполациони полином како бисмо пронашли вредност x, само сада посматрамо вредности $y_j = f(x_j)$ као чворове интерполације. (задатак је на следећој страни)

Нека је дата таблично задата функција $f(x)$

x	2	2.5	3.5	4.0
$y = f(x)$	0.9093	0.5985	−0.3508	−0.7568

Нула таблично дате функције $f(x)$, уколико постоји, заокружена на две децимале је: (помоћ: да би добили тачан резултат, међурезултате рачунајте на 4 децимале) [**3 поена**]

а) 3.14; б) 3.15; в) 3.16; г) нема нуле;

(в) У овом делу разматраћемо технику оптимизације. Циљ оптимизације је пронаћи екстремну тј. најмању/највећу вредност функције циља на неком скупу. Скуп на коме посматрамо функцију циља могуће је дефинисати функцијама ограничења.

Екстремну вредност функције $f(x_1, x_2)$ одређујемо тако што прво налазимо пар (x_1, x_2) за који важе дата ограничења (таквих тачака може бити пуно), а након тога желимо да вредност саме функције циља у тој тачки буде максимална.

Пронаћи вредност решења оптимизационог задатка на скупу **целих бројева**

$$\max f(x_1, x_2) \text{ где } f(x_1, x_2) = -x_1 + 4x_2$$

при ограничењима [**3 поена**]

$$\begin{aligned}
&-10x_1 + 20x_2 \leq 22 \\
&5x_1 + 10x_2 \leq 49 \\
&x_1 \leq 5 \\
&x_1, x_2 \geq 0
\end{aligned}$$

а) 7; б) 8.2; в) 8; г) 6;

Задатак 5. - Геометрија

У геометрији, потенција тачке говори о удаљености неке тачке од неке кружнице. Ако нека права која садржи тачку P сече круг $k(O, r)$ у тачкама A и B, тада је *потенција тачке* P у односу на круг k дата са $p(P, k) = \overrightarrow{PA} \cdot \overrightarrow{PB} = PO^2 - r^2$ и не зависи од избора праве. ($\overrightarrow{PA} \cdot \overrightarrow{PB}$ је стандардни скаларни производ вектора $\overrightarrow{PA}$ и $\overrightarrow{PB}$). Потенција тачке P је већа, мања или једнака нули у зависности од тога да ли се тачка налази у спољашњости, унутрашњости круга или на кругу.

Геометријско место тачака равни које имају једнаке потенције у односу на два круга $k_1(O_1, r_1)$ и $k_2(O_2, r_2)$ је права која се зове *радикална* или *потенцијална оса*.

Радикална оса је управна на правој O_1O_2. Радикалне осе три круга неке равни су или паралелне или се секу у једној тачки. Уколико се секу, пресечну тачку називамо *радикалним центром* тих кругова.

Нека је k кружница описана око оштроуглог троугла ABC, $(AC < BC)$. Симетрала угла $\angle ACB$ сече праву AB у тачки L, тачка M је средиште лука AB кружнице k на којем се налази и тацка C, а I је центар уписане кружнице троугла ABC. Кружница k сече по други пут праву MI у тачки K и кружницу са пречником CI у H. Кружница описана око троугла CLK сече AB поново у T.

(а) Нека симетрала угла $\angle ACB$ сече k у тачки N. Одредити $\angle NKC + \angle CKT$. [**3 поена**]

(б) Доказати $NA = NB = NI$. Затим показати $\triangle NAL \sim \triangle NAC$ и $\triangle NIT \sim \triangle NKI$. [**7 поена**]

(в) Нека је k_1 кружница описана око троугла ABI и k_2 кружница са пречником CI. Одредити радикални центар кружница k, k_1 и k_2, па доказати да су тачке T, H и C колинеарне. [**4 поена**]

СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
HS Algorithm 2022

шифра

Задатак 6. - Алгоритми и структуре података

Стек представља структуру података базирану на принципу LIFO (Last In First Out). Две основне операције за рад са стеком су push и pop. Операцијом push додаје се елемент на врх стека, док се операцијом pop скида елемент са врха и, зависно од имплементација, враћа који је то елемент био.

Пример: Тренутни садржај стека је: 1 2 3 4 5, где је 1 елемент који је на врху. Ако операцијом push додамо број 6 на врх стека, његов садржај ће бити: **6** 1 2 3 4 5, ако затим применимо операцију pop, број 6 се уклања са врха стека и његов садржај постаје 1 2 3 4 5, ако бисмо поново применили операцију pop, стек би изгледао овако: 2 3 4 5.

Начин записивања израза у математици на који смо навикли подразумева да се знак операције налази између два операнда (примери: $2+4$, $2*(3+4)$)) и назива се инфиксна нотација. Постфиксна нотација је начин записивања израза у коме се знак операције наводи после операнада (претходни примери постфиксном нотацијом били би записани као $24+$ и $234+*$. Вредност постфиксно записаних израза једноставно се израчувана коришћењем стека. Када у изразу наиђемо на број (једноставности ради, подразумеваћемо да су бројеви **једноцифрени** и у примеру и у задатку који следи), постављамо га на врх стека, на њих примењујемо оператор, скидамо два броја са врха стека, на њих примењујемо оператор и резултат додајемо на врх стека. Пример: ако нам је дат постфиксно записан израз $234+*$, на стек се прво додају 2 и 3, па је садржај стека **3 2**, затим наилазимо на број 4 и њега додајемо на врх стека чији је садржај сада **4 3 2**, следећи у изразу је плус, па скидамо 4 и 3 са врха стека, израчунавамо вредност израза $4+3$ и додајемо на врх стека, па сада на стеку имамо: **7 2**. Даљим проласком кроз израз наилазимо на операнд $*$, скидамо 7 и 2 са врха стека, множимо их и добијамо коначан резултат 14 који постављамо на врх стека.

Пред вама је израз написан у постфиксној нотацији

$$257+*83*+346*++$$

(a) Репрезентација датог израза у инфиксној нотацији је? [**4 поена**]

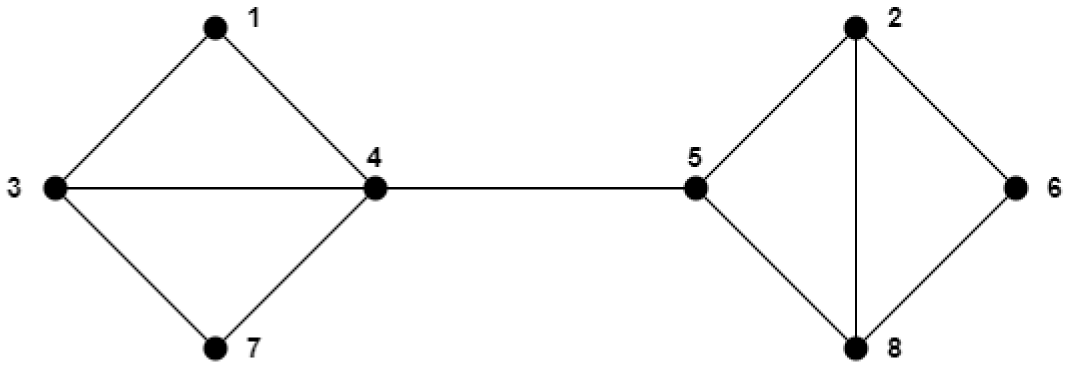
а) $2+5*7+8*3+3*(4+6)$
б) $(2+5)*7+8*3+(3*4+6)$
в) $2*(5+7)+8*3+(3+4*6)$
г) $2*(5+7)+8*3*(3+4+6)$

(б) Збир бројева који се налазе на стеку након 10. операције извршене над стеком је? (под операцијом над стеком у овом контексту мислимо на додавање броја на врх стека или на скидање два броја и додавање њиховог збира тј. производа на врх стека - 3 4 5 → 2 3 4 5 је једна операција над стеком као и 3 4 5 → 12 5) [**5 поена**]

а) 48; б) 49; в) 50; г) 51;

Задатак 7. - Конструкција и анализа алгоритама

Прост граф или само граф је уређени пар $G = (V, E)$ који се састоји од скупа V чије елементе називамо чворови и скупа E који чине двочлани подскупови скупа чворова, а чије елементе називамо гране. Гране се углавном састоје из два чвора, мада постоје и гране чија су оба краја исти чвор (петље). Граф може бити усмерен и неусмерен. Код усмерених грана, гране су представљене као уређени парови чворова тј. одређени су почетак и крај гране - ове гране цртамо стрелицама у правцу усмерења (од почетка ка крају), док се код неусмерених графова, гране цртају обичним линијама. Графови представљају јако важне структуре података у информатици и користе се за решавање многих проблема. Репрезентација графова у рачунарству је најчешће уз помоћ листа повезаности. Сваком чвору се придружује листа чворова до којих постоји директна грана из тог чвора. За два чвора кажемо да су повезани уколико постоји пут између њих. За граф кажемо да је повезан уколико су сви чворови тог графа повезани (постоји пут између било која два чвора а не нужно грана). Повезаност чворова је једна релација еквиваленције дефинисана на графу. Класе еквиваленције у односу на ту релацију називамо компонентама повезаности (нпр. постојале би две компоненте повезаности када би се са графа са слике уклонила грана (4, 5)). Основа за сваки алгоритам са графовима је обилазак графа. Две најпознатије врсте обилазака су у облизак у ширину (BFS облизак) и у дубину (DFS). Оба алгоритма гарантују да ће сваки чвор бити посећен уколико је полазни граф повезан.



(а) Који од следећих понуђених листа повезаности одговара графу са слике: **[3 поена]**

а) $1 \to 3, 4$
$2 \to 5, 6$
$3 \to 1, 4, 7$
$4 \to 1, 3, 7, 5$
$5 \to 4, 2, 8$
$6 \to 2, 8$
$7 \to 3, 4$
$8 \to 5, 2, 6$

б) $1 \to 3, 4$
$2 \to 5, 8, 6$
$3 \to 1, 4, 7$
$4 \to 1, 3, 7, 5$
$5 \to 4, 2, 8$
$6 \to 2, 8$
$7 \to 3, 4, 1$
$8 \to 5, 2, 6$

в) $1 \to 3, 4$
$2 \to 5, 8, 6$
$3 \to 1, 4, 7, 5$
$4 \to 1, 3, 7$
$5 \to 4, 2, 8$
$6 \to 2, 8$
$7 \to 3, 4$
$8 \to 5, 2, 6$

г) $1 \to 3, 4$
$2 \to 5, 8, 6$
$3 \to 1, 4, 7$
$4 \to 1, 3, 7, 5$
$5 \to 4, 2, 8$
$6 \to 2, 8$
$7 \to 3, 4$
$8 \to 5, 2, 6$

(б) **Стек** представља структуру података базирану на принципу LIFO (Last In First Out). Две основне операције за рад са стеком су push и pop. Операцијом push додаје се елемент на врх стека, док се операцијом pop скида елемент са врха и, зависно од имплементација, враћа који је то елемент био.

Пример: Тренутни садржај стека је: 1 2 3 4 5, где је 1 елемент који је на врху. Ако операцијом push додамо број 6 на врх стека, његов садржај ће бити: **6** 1 2 3 4 5, ако затим применимо операцију pop, број 6 се уклања са врха стека и његов садржај постаје 1 2 3 4 5, ако бисмо поново применили операцију pop, стек би изгледао овако: 2 3 4 5.

Кораци алгоритма DFS обиласка су следећи:

(а) Креирати празан стек и низ посећењих чворова

(б) Чвор из кога креће претрага додати у стек

(в) Врти се петља све док стек није празан

(г) Са врха стека се скида један цвор и исписује се

(д) За тај чвор се гледа сваки суседан чвор из листе повезаности и ако није посећен додаје се и он у стек

Псеудо-код је следећи:

```
ulaz: graf sa V cvorova
dfs(cvor){
   posecen: niz gde su sve vrednosti postavljene na False

   stek: stack prazan stek

   stek.push(cvor) dodat je pocetni cvor u stek

   sve dok stek nije prazan radi:

      trenutniCvor = stek.top() uzima se cvor sa vrha steka
      stek.pop() izbacuje sa steka

      ako trenutni cvor nije posecentrenutni Cvor:
         ispisi trenutniCvor
         oznaci cvor kao posecen

      za svakog suseda iz liste povezanosti trenutnog cvora radi:
         ako nije posecen:
            dodaj suseda u stek
```

На основу описа алгоритма и листе повезаности из дела (а) тачан редослед којим ће бити исписани чворови графа са слике уколико претрагу почињемо из чвора са редним бројем 1 је: [**3 поена**]

а) 1 3 4 7 5 2 8 6   б) 1 3 7 4 5 8 6 2   в) 1 4 5 8 6 2 7 3   г) 1 3 7 4 5 2 8 6

(в) Заокружити тачну тврдњу [**3 поена**]

а) Број компонената повезаности неусмереног графа се може израчунати бројањем покретања DFS (покретањем DFS алгоритма из произвољног чвора графа, затим поновним покретањем DFS алгоритма из једног од непосећених чворова све док не буду сви посећени)

б) Уколико је граф усмерен, DFS ће обићи све чворове независно од тога из ког чвора је покренут

в) Уколико је граф повезан и неусмерен, DFS неће обићи све чворове независно од тога из ког чвора је покренут

г) Уколико у графу постоји грана која је петља, DFS алгоритам неће радити исправно

Задатак 8. - Релационе базе података

Релационе базе података представљају модел података који описује начине представљања података, гледања на податке, као и правила за рад са тим подацима. Сви подаци у бази се кориснику приказују искључиво у облику табела. Све табеле задовољавају извесна ограничења. На пример, свака табела мора да има примарни кључ – атрибут (колона) или комбинација атрибута која јединствено идентификује табелу. Оператори који су на располагању корисницима за обраду табела су такви да је њихов резултат увек нова табела.

Вредност у табели може бити и NULL, што означава да поље нема вредност, односно да она није дефинисана. У наставку дајемо пример једне табеле у оквиру релационе базе.

Rbroj_U_Dnevniku	Ime	Prezime	Datum_Rodjenja
1	Vladimir	Andjelic	11.10.2004.
2	Jovana	Guduric	22.08.2004.
3	Ana	Jankovic	10.06.2004.
4	Nikola	Petrovic	24.08.2004.
5	Uros	Stefanovic	02.06.2004.

SQL је стандарни језик за складиштење, манипулацију и испитивање података из базе података. Пример SQL упита је у наставку:

```
SELECT Ime, Prezime
FROM dnevnik
WHERE Ime = "Ana";
```

У датом упиту из табеле 'dnevnik' издвајамо сва имена и презимена из редова у којим је вредност имена једнака 'Ana'.

У команди SELECT ime, prezime одређујемо које колоне издвајамо.
У команди FROM dnevnik одређујемо из које табеле (или табела) издвајамо колоне.
У команди WHERE ime="Ana" задајемо услов издвајања.

Често је неопходно да спојимо две табеле. Покажимо то примером.

tabela proseci

Rbroj_U_Dnevniku	Prosek
1	3.55
2	NULL
6	5
7	4.91
8	2.82

tabela imena

Rbroj_U_Dnevniku	Ime_I_Prezime
1	Vladimir Andjelic
2	Jovana Guduric
5	Ana Jankovic
8	Nikola Petrovic
9	Uros Stefanovic

Команда JOIN спаја две табеле преко заједничке колоне. После наредбе ON наводимо услов спајања. Погледајмо резултат следећег упита.

SELECT proseci.Rbroj_U_dnevniku, Prosek, Ime_I_Prezime
FROM proseci JOIN imena ON proseci.Rbroj_U_Dnevniku = imena.Rbroj_U_Dnevniku;

Rbroj_U_Dnevniku	Prosek	Ime_I_Prezime
1	3.55	Vladimir Andjelic
2	NULL	Jovana Guduric
8	2.82	Nikola Petrovic

Команда LEFT JOIN враћа све записе из леве табеле и одговарајуће записе из десне табеле. Резултат је NULL са десне стране ако нема подударања.

SELECT proseci.Rbroj_U_dnevniku, Prosek, Ime_I_Prezime
FROM proseci LEFT JOIN imena ON proseci.Rbroj_U_Dnevniku = imena.Rbroj_U_Dnevniku;

Rbroj_U_Dnevniku	Prosek	Ime_I_Prezime
1	3.55	Vladimir Andjelic
2	NULL	Jovana Guduric
6	5	NULL
7	4.91	NULL
8	2.82	Nikola Petrovic

Могуће је спојити и више од две табеле. То радимо тако што редом извршавамо спајања.

Дате су табеле:

tabela city

Id	City_Name	Lat	Long	Country_Id
1	Berlin	52.520008	13.404954	1
2	Belgrade	44.787197	20.457273	2
3	Zagreb	45.815399	15.966568	3
4	New York	40.730610	-73.935242	4
5	Los Angeles	34.052235	-118.243683	4
6	Warsaw	52.237049	21.017532	5

tabela customer

Id	Customer_Name	City_Id	Customer_Address	Next_Call_Date	Ts_Inserted
1	Jewelry Store	4	Long Street 120	2020-01-22	2020-01-09 14:01:20.000
2	Bakery	1	Kurfurstendamm 25	2020-02-22	2020-01-09 17:52:15.000
3	Cafe	1	Tauentzienstrasse 44	2020-01-21	2020-01-10 08:02:49.000
4	Restaurant	3	Nikole Tesle 15	2020-01-21	2020-01-10 09:20:21.000

tabela country

Id	Country_Name	Country_Name_Eng	Country_Code
1	Deutchland	Germany	DEU
2	Srbija	Serbia	SRB
3	Hrvatska	Croatia	HRV
4	United States of America	United States of America	USA
5	Polska	Poland	POL
6	Espana	Spain	ESP
7	Rossiya	Russia	RUS

Резултат извршавања упита

```
SELECT country.Country_Name_Eng, city.City_Name, customer.Customer_Name
FROM country
LEFT JOIN city ON city.CountryId = country.Id
LEFT JOIN customer ON customer.City_Id = city.Id;
```

je? **[9 поена]**

а)

Country_Name_Eng	City_Name	Customer_Name
Croatia	Zagreb	Restaurant
Germany	Berlin	Bakery
Germany	Berlin	Cafe
United States of America	New York	Jewelry Store

б)

Id	Country_Name_Eng	City_Name	Customer_Name
3	Croatia	Zagreb	Restaurant
1	Germany	Berlin	Bakery
1	Germany	Berlin	Cafe
4	United States of America	New York	Jewelry Store

в)

Id	Country_Name_Eng	City_Name	Customer_Name
4	Croatia	Zagreb	Restaurant
2	Germany	Berlin	Bakery
3	Germany	Berlin	Cafe
1	United States of America	New York	Jewelry Store

r)

Country_Name_Eng	City_Name	Customer_Name
Croatia	Zagreb	Restaurant
Germany	Berlin	Bakary
Germany	Berlin	Cafe
Poland	Warsaw	NULL
Russia	NULL	NULL
Serbia	Belgrade	NULL
Spain	NULL	NULL
United States of America	New York	Jewelry Store
United States of America	Los Angeles	NULL

Задатак 9. - Вештачка интелигенција

Аутоматско расуђивање је подобласт вештачке интелигенције која се ослања на математичку логику у развијању алгоритама и програма који су способни да аутоматски расуђују. Неки од проблема који се могу решити овим методом су: да ли неко тврђење важи за све објекте, да ли постоји објекат за који важи дато тврђење итд.

Логика првог реда представља 'апарат' који се користи у оквиру аутоматског расуђивања, а централни проблеми који се решавају су испитивање да ли је дата формула таутологија и да ли постоји валуација у којој формула има вредност *тачно.* Постоје програми (решавачи) за чије је коришћење потребно само прецизно формулисати проблем у терминима логике првог реда, а они га затим решавају коришћењем метода аутоматског расуђивања. У писању формула користимо: скуп променљивих, логичке константе $\top$ (тачно) и $\bot$ (нетачно), логичке везнике: негацију, коњункцију, дисјункцију, импликацију и еквиваленцију, као и два квантификатора: универзални ($\forall$ - за сваки) и егзистенцијални ($\exists$ - постоји). Поред наведених компоненти, у примерима и у задатку користићемо и предикатске (релацијске) симболе, нпр. covek(x) значи 'x је човек', lep(dan) значи 'дан је леп', jede(Ana, jabuka) значи 'Ана једе јабуку'. Пример записивања проблема у терминима логике првог реда:

Сваки пас је добар.

Уводимо предикатске симболе:
p(x) : x *је пас*
d(x) : x *је добар*

Решење: $(\forall x)(p(x) \implies d(x))$

Међутим, да би програм (у овом случају као пример користимо програм Vampire) разумео нашу формулу, морамо је превести у погодан формат. Користе се следеће ознаке: уместо $\forall$ користи се !, нпр. 'за свако x' пишемо ![X] :, 'за свако x и y' пишемо ![X, Y] :, уместо $\exists$ користимо ?, па слично пишемо 'постоје x и y' као ?[X, Y] :, Ознаке логичких оператора су:

- Коњункција &
- Дисјункција |
- Негација ~
- Импликација =>
- Еквиваленција <=>

Претходни пример бисмо у овој форми написали на следећи начин

![X] : (p(X) => d(X))

Ваш задатак је да за сваку од реченица датих у наставку одредите који запис јој одговара:

(a) Златни и сребрни накит је вредан. **[3 поена]**

а) ![X] : (vredan(X) => (zlato(X) | srebro(X)))
б) ![X] : ((zlato(X) & srebro(X)) => vredan(X))
в) ?[X] : ((zlato(X) & srebro(X)) => vredan(X))
г) ![X] : ((zlato(X) | srebro(X)) => vredan(X))

(б) Нису сви кишни дани хладни. **[3 поена]**

а) ![X] : (kisa(X) & ∼ hladan(X))

б) ![X] : (∼ kisa(X) => hladan(X))

в) ?[X] : (kisa(X) & ∼ hladan(X))

г) ?[X] : (∼ kisa(X) => hladan(X))

(в) Тигрови и лавови нападају ако су гладни или угрожени. **[3 поена]**

а) ![X] : ((tigar(X) & lav(X)) => ((gladan(X) | ugrozen(X)) => napada(X)))

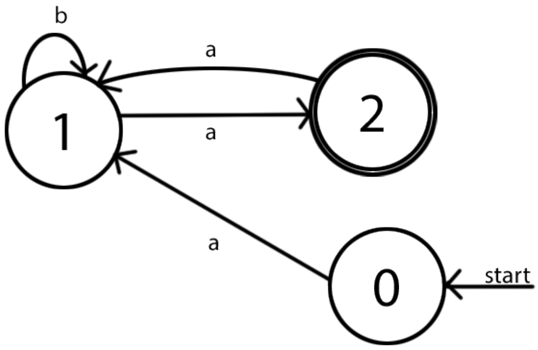
б) ![X] : ((tigar(X) | lav(X)) => ((gladan(X) | ugrozen(X)) => napada(X)))

в) ![X] : ((tigar(X) & lav(X)) => (napada(X) => (gladan(X) | ugrozen(X)))

г) ![X] : ((tigar(X) | lav(X)) => ((gladan(X) | ugrozen(X)) & napada(X)))

Задатак 10. - Превођење програмских језика

Коначни аутомати описују формализам помоћу којег можемо проверити да ли нека реч припада одређеном регуларном језику. Илуструјмо функционисање коначних аутомата следећим примером:



Ово је пример једног детерминистичког коначног аутомата (ДКА). Њему одговара регуларни израз $(ab{*}a)+$. Регуларни изрази представљају "шаблон" по којем се претражују стрингови. У регуларним изразима симбол $*$ означава да се израз понавља 0 или више пута, док симбол $+$ означава да се израз понавља 1 или више пута. Симболи се односе само на последњи карактер, а ако желимо да се односи на више карактера, потребно је да их ставимо у заграду. На пример, регуларни израз $ab*$ ће пронаћи стринг abb, док неће пронаћи стринг $abab$. Стринг $abab$ може да се опише изразом $(ab)*$.

Коначни аутомат је дискретни математички модел који се састоји од коначног броја стања (при чему се унутрашња стања означавају кругом, а завршна стања дуплим кругом), прелаза између тих стања и акција које обавља. Почетно стање је означено стрелицом која води из ознаке start.

Коначне аутомате употребљавамо како бисмо проверили да ли одређени стринг припада језику, односно да ли одговара одређеном регуларном изразу.

Применимо овај аутомат над стрингом $abaaa$ како бисмо проверили да ли припада језику. Крећемо из почетног стања 0, наилазимо на карактер a који читамо и прелазимо у стање 1. Након тога, наилазимо на карактер b који читамо и прелазимо опет у стање 1. Након тога, наилазимо на карактер a који читамо и прелазимо у стање 2. У наставку читамо a, враћамо се у стање 1, читамо a и прелазимо у стање 2. Пошто смо завршили у стању 2, онда можемо да прихватимо задату реч јер је стање 2 завршно.

Применимо исти аутомат над ниском aaa. Крећемо из почетног стања 0, наилазимо на карактер a који читамо и прелазимо у стање 1, затим читамо a и прелазимо у стање 2, поново читамо карактер a и враћамо се у стање 1, али како смо завршили у стању 1, а то стање није завршно стање, задата реч не може бити прихваћена.

Нацртати коначни аутомат за регуларни израз (детаљно образложити одговор) **[14 поена]**

$$aab{*}(ab){+}a$$