СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
А категорија
HS Algorithm 2023

шифра

Број бодова по задацима

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

Укупан број поена:

СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
А категорија
HS Algorithm 2023

шифра

Задатак 1. Диференцијалне једначине

Диференцијалне једначине су једначине које описују понашање различитих физичких, хемијских и других процеса, као што су кретање тела, раст популације или теорија електромагнетизма. Ова грана математике саставни је део природних и техничких наука, те ће већина вас у будућности имати прилике да се сусретне са диференцијалним једначинама и њиховим применама.

Пре него што формално уведемо појам диференцијалних једначина потребно је да се упознамо са основним концептима математичке анализе, тј. појмовима извода, интеграла и примитивне функције:

- **Извод функције** $F(x)$, у ознаци $F'(x)$, је математичка операција која представља брзину промене функције у одређеној тачки, тј. мери колико се функција $F(x)$ промени када се вредност независне променљиве x промени за врло малу вредност. Извод функције $F(x)$ представља проналажење нове функције $f(x)$ која описује брзину промене оригиналне функције у зависности од промене вредности променљиве x, тј. $f(x) = F'(x) = \frac{dF}{dx}$.
- **Интеграл функције** $f(x)$, у ознаци $\int f(x)\,dx$, је математичка операција која представља супротност изводу. Интеграл функције $f(x)$ представља проналажење нове функције $F(x)$, назване интегрална функција или **примитивна функција**, чији је извод једнак оригиналној функцији, тј. $F'(x) = f(x)$.

Да бисмо израчунали извод или интеграл неке функције, потребно је да применимо правила извода, тј. интеграла у складу са математичким операцијама које су присутне у тој функцији. У таблици испод налазе се изводи и интегали често коришћених функција и основна правила њиховог коришћења:

$f(x)$	$f'(x)$
C	0
$x^a \ (a \neq 0)$	ax^{a-1}
a^x	$a^x \ln a$
e^x	e^x
$\log_a x$	$\frac{1}{x \ln a}$
$\ln x$	$\frac{1}{x}$
$\sin x$	$\cos x$
$\cos x$	$-\sin x$
$\operatorname{tg} x$	$\frac{1}{\cos^2 x}$
$\operatorname{ctg} x$	$-\frac{1}{\sin^2 x}$
$\arcsin x$	$\frac{1}{\sqrt{1-x^2}}$
$\arccos x$	$-\frac{1}{\sqrt{1-x^2}}$
$\operatorname{arctg} x$	$\frac{1}{1+x^2}$
$\operatorname{arcctg} x$	$-\frac{1}{1+x^2}$

$f(x)$	$\int f(x)dx$
a	$ax + C$
x^a	$\frac{x^{a+1}}{a+1} + C,\ a \neq -1$
$\frac{1}{x}$	$\ln \lvert x \rvert + C$
a^x	$\frac{a^x}{\ln a} + C$
e^x	$e^x + C$
$\sin x$	$-\cos x + C$
$\cos x$	$\sin x + C$
$\frac{1}{\cos^2 x}$	$\operatorname{tg} x + C$
$-\frac{1}{\sin^2 x}$	$\operatorname{ctg} x + C$
$\frac{1}{\sqrt{1-x^2}}$	$\arcsin x + C$
$\frac{1}{1+x^2}$	$\operatorname{arctg} x + C$

док се у табели испод налазе нека често коришћена правила:

Особине извода	Особине интеграла
$(Cf(x))' = Cf'(x)$	$\int Cf(x)dx = C\int f(x)dx$
$(f(x) \pm g(x))' = f'(x) \pm g'(x)$	$\int (f(x) \pm g(x))dx = \int f(x)dx \pm \int g(x)dx$
$(f(x) \cdot g(x))' = f'(x)g(x) + f(x)g'(x)$	$\int u(x)v'(x)dx = u(x)v(x) - \int v(x)u'(x)dx$
$\left(\frac{f(x)}{g(x)}\right)' = \frac{f'(x)g(x) - f(x)g'(x)}{g^2(x)}$	$\int f'(g(x)) \cdot g'(x)dx = f(g(x)) + C$
$(f(g(x)))' = f'(g(x))g'(x)$	У претходном реду, функција $g(x)$ је смена и пишемо $g(x) = t$, $dt = g'(x)dx$ и то заменимо у интеграл

Када нам је задатак да пронађемо примитивну функцију дате функције $f(x)$ на интервалу (a, b), тражимо функцију $F(x)$ чији је извод једнак датој функцији $f(x)$ на том интервалу. Као што смо већ навели, овај проблем можемо илустровати следећом једначином:

$$F'(x) = f(x)$$

Како је у једначини функција $F(x)$ дата у облику свог првог извода, ова једначина се назива диференцијална једначина првог реда. По конвенцији, непознату функцију независне променљиве, коју смо до сада означавали са $F(x)$, означаваћемо са $y(x)$ или само y.
Аналогно се могу дефинисати и диференцијалне једначине виших редова :

- Диференцијална једначина II реда: $y'' - y = 0$
- Диференцијална једначина III реда: $y''' - 4xy' + 3x = 0$
- У општем случају диференцијалну једначину записујемо као: $F(x, y, y', ..., y^{(n)}) = 0$

Дакле, **диференцијална једначина** је свака једначина у којој се јавља независна променљива x, непозната фунцкија независне променљиве $y(x)$ и изводи те непознате функције.

Решавање диференцијалне једначине подразумева проналажење свих њених решења.

- Ако функција $\phi(x)$ идентички задовољава диференцијану једначину $F(x, y, y', y^{(n)}) = 0$ на интервалу (a, b), функција $\phi(x)$ је **решење диференцијалне једначине** $F(x, y, y', y^{(n)}) = 0$ на интервалу (a, b).
- Фамилија функција $y = \phi(x, c)$, где је c константа, је **опште решење** диференцијалне једначине $y' = f(x, y)$ ако за сваку тачку (a, b) једначина $b = \phi(a, c)$, где је c слободна променљива, има јединствено решење c_0, тако да је функција $y = \phi(x, c_0)$ решење једначине $y' = f(x, y)$ које задовољава почетни услов $y(a) = b$.
- Свако решење једначине $y' = f(x, y)$ добијено од општег решења тако што се параметру c даје конкретна вредност c_0 је **партикуларно решење** диференцијалне једначине.

Један од најосновнијих типова диференцијалних једначина првог реда је **диференцијална једначина која раздваја променљиве**. Она је дефинисана на следећи начин:

$$y' = f(x)g(y)$$

Где су $f(x)$ и $g(y)$ унапред задате функције. Како бисмо дошли до решења ове једначине, потребно је наметнути услов да је $g(y) \neq 0$ (како бисмо избегли могућност дељења нулом) и записати y' као $y' = \frac{dy}{dx}$. Наша једначина тада има облик:

$$\frac{dy}{dx} = f(x)g(y)$$

Пошто смо претпоставили да $g(y) \neq 0$, једначину можемо записати на следећи начин:

$$\frac{dy}{g(y)} = f(x)dx$$

Затим интегралимо постојећу једначину:

$$\int \frac{dy}{g(y)} = \int f(x)dx$$

Одакле налажењем функција $G(y)$ и $F(x)$, које су примитивне функције за $\frac{1}{g(y)}$ и $f(x)$, добијамо решење диференцијалне једначине при услову $g(y) \neq 0$ које је облика:

$$G(y) = F(x) + c$$

Где је $c \in C$ константа интеграције.
Уколико функција $G(y)$ има инверзну функцију, y можемо изразити као:

$$y(x) = G^{-1}\left(F(x) + c\right)$$

Овим поступком добијамо опште решење диференцијалне једначине која раздваја променљиве.

Пример. Решити диференцијалну једначину $y' = x(1 + y^2)$ која задовољава услов $y(0) = 1$.

За почетак, потребно је да приметимо да је $f(x) = x$, а $g(y) = (1 + y^2)$ и $y' = \frac{dy}{dx}$, па једначину има облик:

$$\frac{dy}{dx} = x(1 + y^2)$$

Како $g(y) \neq 0$ за $\forall y \in \mathbb{R}$, једначину можемо записати на следећи начин:

$$\frac{1}{1+y^2}dy = xdx$$

Када интегралимо ову једначину и прочитамо вредности интеграла из горенаведене таблице, добијамо:

$$\int \frac{1}{1+y^2}dy = \int xdx \quad \Rightarrow \quad \operatorname{arctg} y + c_1 = \frac{x^2}{2} + c_2$$

Како би запис био компактнији, константе интеграције c_1 и c_2 можемо написати као $c = c_2 - c_1$,где је константа $c \in \mathbb{R}$. Дакле, наша једначина облика:

$$\operatorname{arctg} y = \frac{x^2}{2} + c$$

Сада можемо изразити y и тиме добити опште решење полазне диференцијалне једначине:

$$y(x) = \operatorname{tg}\left(\frac{x^2}{2} + c\right)$$

Када узмемо у обзир почетни услов $y(0) = 1$ и убацимо га у добијену једначину, добијамо:

$$1 = \operatorname{tg}\left(\frac{0}{2} + c\right) \quad \Rightarrow \quad 1 = \operatorname{tg} c \quad \Rightarrow \quad c = \operatorname{arctg} 1 = \frac{\pi}{4}$$

Дакле, $c = \frac{\pi}{4}$, па је партикуларно решење диференцијалне једначине које задовољава услов $y(0) = 1$ једнако:

$$y(x) = \operatorname{tg}\left(\frac{x^2}{2} + \frac{\pi}{4}\right)$$

Још један пример диференцијалне једначине првог реда је **линеарна диференцијална једначина**. Она је дефинисана на следећи начин:

$$y' + p(x)y = q(x)$$

Где су $p(x)$ и $q(x)$ унапред задате функције. За почетак, потребно је одредити фактор који се множи са диференцијалном једначином како би се поједноставило решавање који је у случају линеарне диференцијалне једначине I реда дефинисан као:

$$\mu(x) = e^{\int p(x)\,dx}$$

Сада можемо помножити целу диференцијалну једначину овим фактором:

$$\mu(x) \cdot y' + \mu(x) \cdot p(x) \cdot y = \mu(x) \cdot q(x)$$

Примећујемо да се пред нама заправо налази извод производа, па једначину можемо написати у облику:

$$\frac{d(\mu(x) \cdot y)}{dx} = \mu(x) \cdot q(x)$$

Затим интегралимо постојећу једначину:

$$\int \frac{d(\mu(x) \cdot y)}{dx}\,dx = \int \mu(x) \cdot q(x)\,dx \quad \Rightarrow \quad \mu(x) \cdot y = \int \mu(x) \cdot q(x)\,dx + c$$

Где је $c \in C$ константа интеграције.
Сада можемо изразити y и добити опште решење линеарне диференцијалне једначине:

$$y(x) = \frac{1}{\mu(x)} \cdot \left(\int \mu(x) \cdot q(x)\,dx + C\right)$$

Две шоље топле кафе познатих почетних температура $T_1(0) = T_2(0) = T_0$ остављене су да се хладе на собној температури T_e таквој да је $T_e < T_0$. Прва шоља се остави недирнута и промена њене температуре $T_1(t)$ кроз време може се моделирати Њутновим законом хлађења:

$$\frac{dT_1}{dt} = -k(T_1(t) - T_e)$$

Где је $k > 0$ позната константа која зависи од структуре и количине течности у шољи. У другу шољу се од тренутка $t = \frac{1}{2k}$ почне сипати вода константом брзином која је све топлија и топлија (линеарно са временом).

(а) [2 поена] Промена температуре кафе кроз време у другој шољи $\left(\text{за } t \neq \frac{1}{2k}\right)$ може се моделирати једначином:

а) $\frac{dT}{dt} = -k(T - T_e) + \left(\frac{1}{4} + \operatorname{sgn}\left(t - \frac{1}{2k}\right)\right) 2t$ б) $\frac{dT}{dt} = -k(T - T_e) + \left(\frac{1}{2} + \operatorname{sgn}\left(t - \frac{1}{2k}\right)\right) t$

в) $\frac{dT}{dt} = -k(T - T_e) + \left(1 + \operatorname{sgn}\left(t - \frac{1}{2k}\right)\right) \frac{1}{2}t$ г) $\frac{dT}{dt} = -k(T - T_e) + \left(2 + \operatorname{sgn}\left(t - \frac{1}{2k}\right)\right) \frac{1}{4}t$

(б) [**3 поена**] Претпоставимо да је температура непрекидна функција од времена, да су константе $T_0 = 95°C$ и $T_e = 20°C$ и да се време мери у минутима. Колика ће бити температура кафе у другој шољи након 20 минута, уколико је након 1 минута температура кафе у првој шољи била $90°C$?

а) $66,185°C$ б) $82,165°C$ в) $108,105°C$ г) $143,125°C$

(в) [**4 поена**] Посматрајмо сада примену Њутновог закона хлађења на комплекснијем систему. Јаје је у тренутку t стављено у воду температуре $T_e(t)$ (температура воде не мора бити константна). Јаје се састоји из два дела, жуманцета које је окружено беланцетом и беланцета које је са једне стране окружено жуманцетом, а са друге стране водом. Како ова два дела нису истог састава, они ће се хладити/загревати различитом брзином. Претпоставимо да жуманце има температуру $T_1(t)$, а беланце има температуру $T_2(t)$. Уколико су a и b позитивне константе, промена температура жуманцета и беланцета кроз време може се моделирати једначинама:

а) $\frac{dT_1}{dt} = -a(T_2 - T_1)$, $\frac{dT_2}{dt} = -a(T_2 - T_1) - b(T_2 - T_e)$

б) $\frac{dT_1}{dt} = -a(T_1 - T_2)$, $\frac{dT_2}{dt} = a(T_2 - T_1) - b(T_2 - T_e)$

в) $\frac{dT_1}{dt} = a(T_2 - T_1)$, $\frac{dT_2}{dt} = -a(T_2 - T_1) + b(T_e - T_2)$

г) $\frac{dT_1}{dt} = a(T_1 - T_2)$, $\frac{dT_2}{dt} = a(T_1 - T_2) + b(T_e - T_2)$

Задатак 2. Линеарна алгебра и аналитичка геометрија

Један од корисних концепата у геометрији и рачунарској графици представљају **афине трансформације**, односно **афина пресликавања**. То су функције између афиних простора које пресликавају тачке у тачке, праве у праве и равни у равни. Једноставности ради, у овом задатку ћемо се бавити само афиним трансформацијама равни.

Дефиниција [афине трансформације равни]: Афина трансформација равни је непрекидна бијекција $\bar{f} : \mathbb{R}^2 \to \mathbb{R}^2$ која очувава колинеарност тачака.

Афина пресликавања имају згодну матричну репрезентацију. За почетак, објаснимо како можемо помножити две матрице.

Нека је A матрица димензија $m \times n$, а B матрица димензија $n \times p$.

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

тада је производ матрица A и B матрица $C = AB$ димензија $m \times p$:

$$C = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix}$$

таква да важи: $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^{n} a_{ik}b_{kj}$, за све $i = 1, ..., m$ и $j = 1, ..., p$. Другим речима, вредност у i-тој врсти и j-тој колони матрице C добијамо скаларним множењем i-те врсте матрице A и j-те врсте матрице B. Наведимо пример множења матрице димензије 2×4 матрицом димензије 4×3.

$$\begin{pmatrix} \mathbf{2} & \mathbf{-3} & \mathbf{0} & \mathbf{5} \\ 1 & -1 & 2 & 4 \end{pmatrix} \begin{pmatrix} \mathbf{3} & 3 & -1 \\ \mathbf{-2} & -4 & 0 \\ \mathbf{-1} & -3 & -1 \\ \mathbf{2} & -5 & 10 \end{pmatrix} = \begin{pmatrix} \mathbf{22} & -7 & 48 \\ 11 & -19 & 37 \end{pmatrix}$$

У првој врсти и првој колони добијене матрице се налази број 22 зато што је $22 = 2 \cdot 3 + (-3) \cdot (-2) + 0 \cdot (-1) + 5 \cdot 2$.

Нека је дат координатни систем Oe и нека се тачка $M(x_1, y_1)$ пресликава у тачку $M'(x_1', y_1')$ афиним пресликавањем f. Ово пресликавање се може представити у облику

$$\begin{pmatrix} x_1' \\ y_1' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Ово сажето можемо написати у облику $X' = AX + b$, где су X', A, X и b одговарајуће матрице из претходне једнакости. Матрица A назива се *линеарни део*, док се матрица b назива *транслаторни део*. Уколико не бисмо користили матрични запис, формуле афиног пресликавања у равни би имале следећи облик:

$$x_1' = a_{11}x_1 + a_{12}y_1 + b_1$$

$$y_1' = a_{21}x_1 + a_{22}y_1 + b_2$$

Претходне две једнакости се још компактније могу записати у матричном облику ($X' = A_bX$) на следећи начин:

$$\begin{pmatrix} x_1' \\ y_1' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Матрица A_b назива се *проширена матрица* афиног пресликавања. Најзначајније афине трансформације су транслација, ротација, рефлексија, скалирање, хомотетија и смицање. Упознајмо се детаљније са њима.

Транслација $\tau_{\vec{b}}$ за вектор $\vec{b} = (b_1, b_2)$ дата је формулама:

$$x' = x + b_1,$$

$$y' = y + b_2,$$

или у матричном облику:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

Ротација око координатног почетка, за угао $\phi \in [0, 2\pi)$ дата је матричном формулом:

$$R_\phi : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\ \phi & -\sin\ \phi \\ \sin\ \phi & \cos\ \phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Ротација око произвољне тачке $Q = (q_1, q_2)$ за угао ϕ се може представити као композиција две транслације и једне ротације око координатног почетка O:

$$R_{Q,\phi} = \tau_{\overrightarrow{OQ}} \circ R_\phi \circ \tau_{\overrightarrow{QO}}$$

Ово тврђење је познато као *теорема о трансмутацији.*

Рефлексија у односу на праву p_0 која пролази кроз координатни почетак и која са позитивним делом x-осе гради угао $\frac{\phi}{2}$ има следећу формулу:

$$S_{p_0} : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\ \phi & \sin\ \phi \\ \sin\ \phi & -\cos\ \phi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Скалирање у правцу координатних оса, са центром у координатном почетку и коефицијентима $\lambda_1, \lambda_2 \neq 0$ има следећу формулу:

$$H_{\lambda_1,\lambda_2} : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Хомотетија представља специјалан случај скалирања, где је $\lambda_1 = \lambda_2$.

Смицање са коефицијентом λ у правцу x-осе има формулу:

$$S_x(\lambda) : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix},$$

док се смицање са истим коефицијентом у правцу y-осе може представити на следећи начин:

$$S_y(\lambda) : \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ \lambda & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}.$$

Посебно значајну класу афиних трансформација представљају **изометрије**. То су пресликавања која чувају *дужину* у еуклидском простору $\mathbb{E}$ произвољне димензије. Карактеристика свих изометрија је да је детерминанта њихове проширене матрице увек једнака 1 или -1.

Неке од најзначајнијих особина афиних трансформација су:

- пресликавају праве у праве;
- чувају размеру колинеарних дужи;
- чувају паралелност правих;
- однос површина слике и оригинала једнак је $\frac{P(F')}{P(F)} = |det(a_{ij})|$, где је $det(a_{ij})$ детерминанта (проширене) матрице датог афиног пресликавања.

Наведимо још и једну корисну теорему.

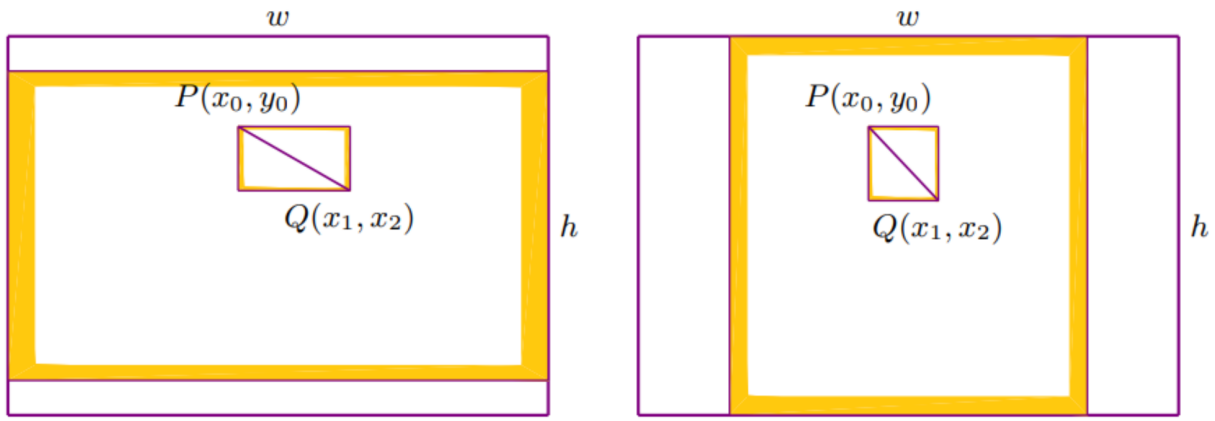
Теорема: Производ проширених матрица афиних пресликавања f_1 и f_2 једнак је проширеној матрици композиције та два пресликавања.

Напомена: Уопштено се афине трансформације могу дефинисати за произвољне векторске просторе (о векторским просторима је било речи на прошлогодишњем такмичењу HS Algorithm). Нека је $\bar{f} : \mathbb{V} \to \mathbb{V}$

линеарно пресликавање векторског простора који је придружен простору тачака $\mathbb{E}$. *Афино пресликавање* $f : \mathbb{E} \to \mathbb{E}$ је пресликавање тачака које је индуковано пресликавањем $\bar{f}$ вектора у смислу да је:

$$f(M) = M', f(N) = N' \iff \bar{f}(\overrightarrow{MN}) = \bar{f}(\overrightarrow{M'N'}).$$

(а) [2 поена] Размотримо пример алатке "*Zoom to window*", приказане на слици испод. Миш је притиснут у тачки $P(360, 420)$, а отпуштен у тачки $Q(520, 520)$. Афиним пресликавањем треба увећати прозор са дијагоналом PQ преко целог екрана. При томе водити рачуна да се увећана слика уклопи у екран или по ширини, или по висини, у зависности од пропорција прозора (без дисторзије). Сматрати да је екран резолуције $1920 : 1080 = 16 : 9$. Колико износи збир свих елемената проширене матрице овог афиног пресликавања?



а) -8328 б) 0 в) 21.6 г) -8305.4

(б) [2 поена] Нека су P и Q произвољне тачке ($P \neq Q$). Дата су четири исказа везана за композиције различитих афиних пресликавања.

1. $R_{Q,\alpha} = \tau_{\overrightarrow{QP}} \circ R_{P,\alpha} \circ \tau_{\overrightarrow{PQ}}$
2. $R_{Q,\alpha} \circ H_{Q,\lambda} = H_{P,\lambda} \circ R_{P,\alpha}$
3. $H_{P,\lambda} = \tau_{\overrightarrow{QP}} \circ H_{Q,\lambda} \circ \tau_{\overrightarrow{PQ}}$
4. $H_{Q,-1,1} \circ H_{Q,1,-1} = R_{Q.\pi}$

Број тачних исказа је:

а) 1 б) 2 в) 3 г) 4

(в) [2 поена] Ротацијом тачке $A(0, \sqrt{3})$ око координатног почетка за 30° добија се тачка B. Рефлексијом у односу на праву која пролази кроз координатни почетак и која са негативним делом x-осе заклапа угао од 45° тачка B се пресликава у тачку C. Хомотетијом са коефицијентом λ се ΔOBC пресликава у $\Delta OB'C'$. За коју вредност параметра λ је тачка $M(-\sqrt{3}, \sqrt{3})$ тежиште $\Delta OB'C'$?

а) $3(\sqrt{3} - 1)$ б) $3(\sqrt{3} + 1)$ в) $\dfrac{3 + \sqrt{3}}{6}$ г) $\sqrt{3}$

(г) [3 поена] Дат је квадрат $ABCD$ са теменима $A(-1, -1)$, $B(1, -1)$, $C(1, 1)$ и $D(-1, 1)$. Афиним трансформацијама он се пресликава у паралелограм $A'B'C'D'$, где су координате тачака $A'(4, 5)$, $B'(8, 7)$ и $C'(6, 9)$. Колика је површина слике круга уписаног у квадрат $ABCD$ при овој трансформацији?

а) π б) 12π в) 3 г) 3π

Задатак 3. Елементи финансијске математике

Најважнији вредносни папири, са приходом који није фиксан, су *акције* (share, stock). Онај ко поседује акције неке фирме је, у одређеном проценту, власник те фирме. Акције имају своју цену, оне се могу куповати и продавати, тј. са њима се може трговати на берзи. *Опције* (options) су такозвани финансијски деривати, односно вредносни папири који се односе на неке друге вредносне папире (обично акције) и чија је цена изведена из цене тих других вредносних папира. Опција је уговор који ономе ко га поседује даје право, али не и обавезу, да купи или да прода неки вредносни папир под одређеним условима. Као и сваки уговор, опција има две стране. У овом случају то су купац и продавац. Тако *кол опција* или куповна опција даје право, али не и обавезу, куповине акција или неких других вредносних папира. *Пут опција* или продајна опција даје право, али не и обавезу, продаје акција или неких других вредносних папира. Оба типа опција имају *уговорену цену* (цена по истеку, цена по доспећу) и *уговорено време* (време истека, време доспећа). Најважнија сврха финансијских деривата је обезбеђење од ризика, односно смањење ризика од губитака. За особу која продаје каже се да је у *краткој позицији*, док особа која купује у *дугој позицији*. *Портфолио* је колекција вредносних папира које неко поседује. Вредност портфолија је линеарна комбинација вредности појединих вредносних папира.

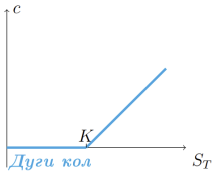
Уговорену цену, по којој се може купити односно продати вредносни папир, обележаваћемо са K, а време до истека опције са T. Почетну цену вредносног папира у тренутку склапања уговора означаваћемо са S_0, а цену у тренутку t са S_t, $0 \leq t \leq T$.

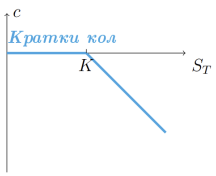
Претпоставимо да имамо кол опцију са временом истека T и уговореном ценом K. Колико вреди кол опција у тренутку T?

- Ако је $S_T < K$, вредност опције је нула јер нећемо куповати акцију за цену K, када на тржишту можемо купити јефтиније за цену S_T.
- Ако важи $S_T > K$, кол опција има вредност јер нам даје могућност да акцију, чија је цена S_T у тренутку T, купимо по нижој уговореној цени K. Активирањем опције купујемо акцију по цени K, коју можемо одмах да продамо по нижој цени S_T и зарадимо профит једнак $S_T - K$.

Дакле, вредност кол опције у тренутку истека T једнака је

$$c = \max\{S_T - K, 0\}.$$



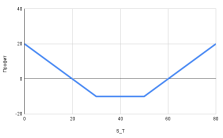


Приметимо да вредност кол опције расте када расте S_T, а опада када расте K. За пут опцију важи обрнуто.

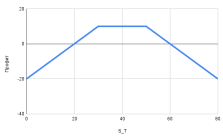
(а) [2 поена] Претпоставимо да опције имају време доспећа T и уговорену цену K. Инвеститор у тренутку T има једну кол и једну пут опцију. Тада је добит инвеститора у тренутку T једнака:

а) $S_T - K$ б) $K - S_T$ в) $|S_T - K|$ г) S_T

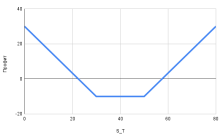
(б) [2 поена] Инвеститор купује кол опцију са уговореном ценом $K_1 = 50$ евра по цени од $c = 4$ евра и пут опцију са уговореном ценом $K_2 = 30$ евра, по цени од $p = 6$ евра. Обе опције имају исто време трајања T. Који од следећих графика приказује зависност профита инвеститора од цене акције у тренутку T:



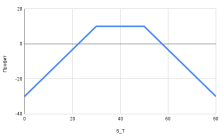
а)



б)



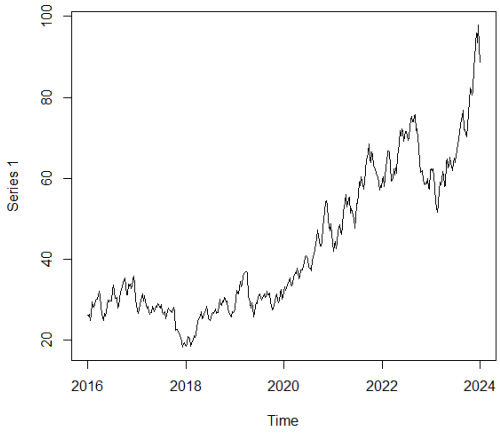
в)



г)

(в) [**2 поена**] На слици испод приказан је график кретања цене акција компаније "СУМА МАТФ" за период од 2016. до 2024. године. Коју позицију треба да заузме инвеститор у децембру 2024. године знајући ове податке:

а) дугу позицију б) кратку позицију в) Није могуће донети одлуку на основу доступних података.



(г) [**3 поена**] За налажење цене европске кол опције полазимо од биномног модела и процене која је неутрална од ризика. Размотримо биномни модел са једним кораком. Садашња вредност опције је S_0, а цену кол опције са истеком T означимо са f. Током живота опције претпоставимо да цена акције може или да порасте за фактор $u > 1$ и да износи uS_0 или да опадне до dS_0 за фактор $d < 1$. Ако цена акције порасте на uS_0, добит од кол опције износи $f_u = \max\{uS_0 - K, 0\}$, а ако цена акције падне до dS_0, добит од кол опције износи $f_d = \max\{dS_0 - K, 0\}$, где је K уговорена цена. Претпоставимо да смо формирали портфолио који се састоји од:

- Δ акција које смо купили и чија је тренутна цена S_0;
- Продали смо једну европску кол опцију на те акције.

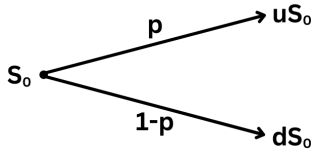
Ако цена акција порасте, вредност портфолиа је $uS_0 - f_u$, а ако цена акција падне у тренутку T вредност портфолиа је $dS_0 - f_d$. Да би портфолио био безризичан, ове две вредности изједначавамо. Из принципа процене која је неутрална од ризика добијамо да је цена опције једнака:

$$f = S_0\Delta - (uS_0\Delta - f_u)e^{-rT},$$

где је r каматна стопа без ризика. Претходна релација се може приказати и на следећи начин

$$f = e^{-rT}(pf_u + (1-p)f_d),$$

где је p изражено преко вредности за u, d, r, T. Овом формулом је дата цена опције у случају када се промена цена акције моделира биномним моделом са једним кораком.



Цена акције је тренутно 45 евра. Познато је да након сваког од два двомесечна периода цена скочи за 10% или падне за 8%. Одредити вредност четворомесечне европске кол опције на ту акцију са уговореном ценом 46 евра. Безризична каматна стопа је 10% годишње. Како је у питању кол опција, акција се активира само у случају када је $S_T > K$. Формирати биномни модел са два корака.

а) 2.44 б) 0.57 в) 2.36 г) 0.64

СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
А категорија
HS Algorithm 2023

шифра

Задатак 4. Алгебра

Алгебра је математичка дисциплина која се бави симболима (које називамо променљивама) и аритметичким операцијама на њима. Комбинацијом променљивих, оператора и константи добијамо математичке изразе. На основу употребе и сложености израза, алгебра се може класификовати у различите гране. Једна од грана алгебре је апстрактна алгебра, а ми ћемо се у наставку бавити једним од централних појмова апстрактне алгебре, а то су **групе**.

Дефиниција. Група је алгебарска структура $(G, *)$, где је G непразан скуп, а $*$ бинарна операција на G која има следеће особине:

1. асоцијативност: За све $x, y, z \in G$ важи $(x * y) * z = x * (y * z)$
2. постојање неутрала: Постоји $e \in G$ такво да за све $x \in G$ важи $x * e = e * x = x$
3. постојање инверза: За свако $x \in G$ постоји $x^{-1} \in G$ такво да је $x * x^{-1} = x^{-1} * x = e$

Ред коначне групе G јесте број елемената скупа G.

Приметимо да је $(\mathbb{Z}, +)$ (где је $+$ стандардна операција сабирања) једна група: сабирање је асоцијативно, неутрал је 0, док је инверзан елемент за $x \in \mathbb{Z}$ елемент $-x \in \mathbb{Z}$.

Сада ћемо се бавити једном конкретном врстом група које називамо **симетричним групама**. Нека је X непразни скуп. Скуп свих бијективних функција које сликају елементе скупа X у елементе скупа X тј.

$$\operatorname{Sym} X = \{f \colon X \to X | f \text{ је бијекција}\}$$

чини групу у односу на композицију пресликавања тј. $(\operatorname{Sym} X, \circ)$ је група. Та група се назива симетричном групом. Група $\operatorname{Sym}\{1, 2, \ldots, n\}$ означавамо са $\mathbb{S}_n$ и називамо кратко симетричном групом или **групом пермутација.** Ред ове групе је $n!$ тј. број бијекција скупа $\{1, \ldots, n\}$ у $\{1, \ldots, n\}$ једнак је $n!$.

На пример, функција f дефинисана са $f(1) = 1$, $f(2) = 2$, $f(3) = 3$ је елемент $\mathbb{S}_3$ и представља неутрал за ту групу. Још један пример елемента $\mathbb{S}_6$ је функција g дефинисана са $g(1) = 3$, $g(2) = 1$, $g(3) = 2$, $g(4) = 5$, $g(5) = 4$, $g(6) = 6$. Елементе скупа $\mathbb{S}_n$ можемо записивати и таблично. Тако претходно дефинисану g можемо записати и на следећи начин

$$g \colon \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 1 & 2 & 5 & 4 & 6 \end{pmatrix}$$

Најпогоднији запис за приказивање елемената групе $\mathbb{S}_n$ су **циклови**. Покажимо на конкретној функцији g како записујемо елементе $\mathbb{S}_n$ као циклове. Крећемо од елемента 1 - видимо да се он слика у 3. Самим тим, отварамо нови цикл $\lceil 1, 3$. Сада гледамо у који се елемент слика 3, то је 2 - продужавамо наш цикл $\lceil 1, 3, 2$. Даље, приметимо да се 2 слика у елемент 1, којим смо започели цикл - самим тим, затварамо цикл и пишемо $\lceil 1, 3, 2\rfloor$. Настављамо од следећег броја који већ није у нашем циклу - то је 4 и поступак понављамо. На крају добијамо $g = \lceil 1, 3, 2\rfloor\lceil 4, 5\rfloor\lceil 6\rfloor$, међутим циклове који садрже један елемент не пишемо па је $g = \lceil 1, 3, 2\rfloor\lceil 4, 5\rfloor$.

(а) [3 поена] Број циклова дужине 5 (што зовемо и 5-цикловима) таквих да се елемент 2 не слика у 3 у групи $\mathbb{S}_{10}$ једнак је

а) 336; б) 2688; в) 5712; г) 6048; н) не знам

Размотримо својства циклова на једноставнијем примеру. Једноставности ради, посматрајмо групу $\mathbb{S}_4$ и два цикла $f = \lceil 1, 3, 4\rfloor$ и $g = \lceil 2, 4\rfloor$. У табличном запису је:

$$f \colon \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 2 & 4 & 1 \end{pmatrix} \quad g \colon \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 4 & 3 & 2 \end{pmatrix}$$

Прво битно својство јесте да чланове цикла можемо померати за произвољан број места удесно. На пример, важи $f = \lceil 1, 3, 4\rfloor = \lceil 4, 1, 3\rfloor = \lceil 3, 4, 1\rfloor$. Како су циклови две пермутације, циклове можемо компоновати - запис $\lceil 1, 3, 4\rfloor\lceil 2, 4\rfloor$ је композиција $f \circ g$ и једнака је $\lceil 2, 1, 3, 4\rfloor$. Свака композиција циклова (а самим тим и

свака пермутација, тј. елемент $\mathbb{S}_n$) може се на сличан начин записати као композиција дисјунктних циклова и тај облик је изузетно погодан за одређивање **реда пермутације**. Прво, треба приметити да дисјунктни циклови комутирају тј. ако су c_1 и c_2 два дисјунктна цикла, тада $c_1 \circ c_2 = c_2 \circ c_1$. На пример, важи $\lceil 1,3 \rfloor \lceil 2,4,5 \rfloor = \lceil 2,4,5 \rfloor \lceil 1,3 \rfloor$. Ако је пермутација f дата као композиција дисјунктних циклова $f = c_1 c_2 \dots c_n$, ред пермутације f је $r(f) = \text{НЗС}(\ell_1, \dots, \ell_n)$ где су $\ell_1, \dots, \ell_n$ редом дужине циклова $c_1, \dots, c_n$. Може се показати да је ред пермутације најмањи природан број такав да је $f^{r(f)} = \underbrace{f \circ f \circ \dots \circ f}_{r(f) \text{ пута}} = e$ где је e неутрал групе $\mathbb{S}_n$

(б) [**3 поена**] Нека је дата пермутација $\pi = \lceil 3,7,1 \rfloor \lceil 7,2,8 \rfloor \lceil 5,4,3 \rfloor \lceil 1,4 \rfloor \in \mathbb{S}_8$. Ред пермутације π^{30} је

а) 1; б) 2; в) 10; г) 12; н) не знам

(в) [**3 поена**] Нека је $A = \{r(f) | f \in \mathbb{S}_8\}$ скуп свих могућих редова пермутација из $\mathbb{S}_8$. Збир елемената из A је?

а) 120; б) 73; в) 36; г) 8; н) не знам

СТУДЕНТСКА УНИЈА МАТЕМАТИЧАРА

Финално такмичење
А категорија
HS Algorithm 2023

шифра

Задатак 5. Математичка анализа

Један од основних појмова у Математичкој анализи јесте **интеграл**. Кроз историју развоја математике, различити научници дали су више различитих дефиниција интеграла - међу којима се издвајају два: Риман (Riemann) и Лебег (Lebesgue). Риманова дефиниција интеграла је класичнија, јаснија и разумљивија али има одређене недостатке које Лебегова дефиниција превазилази, али су теоријске основе за дефинисање Лебеговог интеграла доста комплексније и почињу од дефинисања **алгебри** и **σ-алгебри** (σ - грчко слово сигма) скупова као и функције **мере** на тим алгебрама.

Нека је дат произвољан скуп X (примера ради, нека $X = \mathbb{R}$). **σ-алгебра** $\mathfrak{M}$ скупова на X јесте скуп скупова такав да важи:

1. $\varnothing \in \mathfrak{M}$ - празан скуп увек припада σ-алгебри
2. $E \in \mathfrak{M} \implies E^C = X \setminus E \in \mathfrak{M}$ - ако скуп E припада σ-алгебри, тада и његов комплемент $X \setminus E$ мора припадати σ-алгебри
3. $E_1, \ldots, E_n, \ldots \in \mathfrak{M} \implies E_1 \cup \ldots \cup E_n \cup \ldots \in \mathfrak{M}$ (бесконачна унија скупова из $\mathfrak{M}$ јесте скуп који припада $\mathfrak{M}$)

За $X = \mathbb{R}$, један пример σ-алгебре на X био би скуп: $\mathfrak{M} = \{\varnothing, (-\infty, 3), [3, +\infty), \mathbb{R}\}$.

Једно од битнијих својства σ-алгебри јесте да свака коначна σ-алгебра мора садржати 2^n елемената за неки природан број $n \in \mathbb{N}$. Претпоставимо сада да је дат неки произвољан скуп подскупова од X - (нпр. нека је дат $\mathcal{I} = \{(-\infty, 3)\}$). Минимална σ-алгебра генерисана са $\mathcal{I}$ јесте σ-алгебра $\mathfrak{M}_{\mathcal{I}}$ која има најмањи могући број елемената, а садржи све елементе из $\mathcal{I}$ - у нашем случају то је $\mathfrak{M}_{\mathcal{I}} = \{\varnothing, (-\infty, 3), [3, +\infty), \mathbb{R}\}$.

(а) [4 поена] Нека је $X = \mathbb{Q}_+$ скуп свих позитивних рационалних бројева. Дати су скупови A_0, A_1, A_2, A_3 на следећи начин:

$$A_0 = \{\tfrac{p}{q} \in \mathbb{Q}_+ \mid \text{НЗД}(p, q) = 1,\ p + q \equiv 0 \pmod 4\}$$

$$A_1 = \{\tfrac{p}{q} \in \mathbb{Q}_+ \mid \text{НЗД}(p, q) = 1,\ p + q \equiv 1 \pmod 4\}$$

$$A_2 = \{\tfrac{p}{q} \in \mathbb{Q}_+ \mid \text{НЗД}(p, q) = 1,\ p + q \equiv 2 \pmod 4\}$$

$$A_3 = \{\tfrac{p}{q} \in \mathbb{Q}_+ \mid \text{НЗД}(p, q) = 1,\ p + q \equiv 3 \pmod 4\}$$

Одредити минималне σ-алгебре $\mathfrak{M}_1$ генерисану скуповима A_0, A_2 и $\mathfrak{M}_2$ генерисану скуповима A_0, A_2, A_3.

Основна идеја при дефинисању алгебри скупова јесте одредити фамилију (тј. скуп) скупова које можемо **мерити**. **Мера** на σ-алгебри $\mathfrak{M}$ јесте функција $\mu \colon \mathfrak{M} \to [0, +\infty]$ за коју важи:

1. $\mu(\varnothing) = 0$ - мера празног скупа је 0
2. Нека су $E_1, E_2, \ldots \in \mathfrak{M}$ дисјунктни скупови. Тада

$$\mu\left(\bigsqcup_{n=1}^{\infty} E_n\right) = \sum_{n=1}^{\infty} \mu(E_n)$$

где је $\bigsqcup_{n=1}^{\infty} E_n = E_1 \cup E_2 \cup \ldots$ бесконачна дисјунктна унија и $\sum_{n=1}^{\infty} \mu(E_i) = \mu(E_1) + \mu(E_2) + \ldots$ бесконачни збир. (Напомена: може се показати да дати бесконачни збир увек постоји и вредност таквог збира може бити елемент скупа $[0, +\infty]$ - ово је специфично за редове са позитивним члановима)

Битно је напоменути да се при рачунању са мерама узима да је $+\infty + a = +\infty$ где $a \in (-\infty, +\infty]$ као и $+\infty \cdot a = +\infty$ за $a \in (0, +\infty]$ док важи $+\infty \cdot 0 = 0$ (што није случај у осталим деловима Математичке анализе - $+\infty \cdot 0$ је нешто што називамо неодређеном формом).

(б) [2 поена] Нека је дата σ-алгебра

$$\mathfrak{M} = \{\varnothing, (-\infty, 3), [3, 5), [5, +\infty), (-\infty, 5), [3, +\infty), (-\infty, 3) \cup [5, +\infty), \mathbb{R}\}$$

и нека је дата мера $\mu\colon \mathfrak{M} \to [0, +\infty]$ таква да $\mu((-\infty, 3)) = 1$, $\mu([3, 5)) = 2023$, $\mu(\mathbb{R}) = +\infty$. Одредити мере $\mu([5, +\infty))$ и $\mu((-\infty, 5))$.

Може се показати да за меру $\mu\colon \mathfrak{M} \to [0, +\infty]$ важи следеће својство: ако $A, B \in \mathfrak{M}$ и важи $A \subseteq B$ тада $\mu(A) \leq \mu(B)$. Размотримо сада произвољан скуп $E \in \mathfrak{M}$ такав да $\mu(E) = 0$. Тада за све $N \in \mathfrak{M}$ такве да $N \subseteq E$ важи $\mu(N) = 0$. Међутим, може се десити да ниједан подскуп од E не припада σ-алгебри $\mathfrak{M}$, а било би пожељно да можемо мерити скупове чије мере можемо одредити (на неки логичан начин). Зато уводимо појам **комплетне мере**. Мера $\mu\colon \mathfrak{M} \to [0, +\infty]$ је **комплетна** уколико за сваки скуп $E \in \mathfrak{M}$ такав да $\mu(E) = 0$ важи да и сви $N \subseteq E$ припадају $\mathfrak{M}$ тј. $N \in \mathfrak{M}$.

(в) [8 поена] Дата је σ-алгебра $\mathfrak{M}$ и мера $\mu\colon \mathfrak{M} \to [0, +\infty]$ која није комплетна. Конструисати σ-алгебру $\overline{\mathfrak{M}}$ и меру $\overline{\mu}$ тако да је $\overline{\mu}$ комплетна мера.

Задатак 6. Конструкција и анализа алгоритама

Сегментно стабло, познато и као стабло распона, је структура података која се користи за ефикасно извођење упита распона на низу елемената. Када се конструише сегментно стабло за дат низ, можете брзо израчунати одговоре на различите упите, као што су минимум, максимум, збир или производ елемената унутар одређеног распона.

Најједноставнија имплементација подразумева да се стабло чува имплицитно у низу. Претпоставићемо да елементе стабла смештамо од позиције 1, јер је тада аритметика са индексима мало једноставнија.

(а) [5 поена] Нека је дат низ бројева $a = [3, 5, 2, 5, 9, 10, 7, 2, 4, 11, 6, 2, 9, 8, 12, 0]$. Треба конструисати сегментно стабло примењујући наредни алгоритам:

```
// од елемената низа a са позиција [x, y]
// формира се сегментно стабло и елементи му се
// смештају у низ stablo кренувши од позиције k

void formirajSegmentnoStablo(int a[], int stablo[], int k, int x, int y) {
    if (x == y)
        stablo[k] = a[x];
    else {
        int s = (x + y) / 2;
        formirajSegmentnoStablo(a, stablo, 2*k, x, s);
        formirajSegmentnoStablo(a, stablo, 2*k+1, s+1, y);
        stablo[k] = stablo[2*k] + stablo[2*k+1];
    }
}

// на основу датог низа a дужине n
// у ком су елементи смештени од позиције 0
// формира се сегментно стабло у ком се елементи
// смештају у низ stablo кренувши од позиције 1

void formirajSegmentnoStablo(int a[], int n, int stablo[]) {

    // крећемо формирање од корена који се налази у
    // низу stablo на позицији 1 и покрива елементе
    // на позицијама [0, n-1]

    formirajSegmentnoStablo(a, stablo, 1, 0, n-1);
}
```

Заокружи тачан одговор:

а) $stablo = [-, 95, 43, 52, 15, 28, 23, 29, 8, 7, 19, 9, 15, 8, 17, 12, 3, 5, 2, 5, 9, 10, 7, 2, 4, 11, 6, 2, 9, 8, 12, 0]$

б) $stablo = [-, 95, 43, 52, 15, 8, 17, 28, 23, 29, 8, 7, 19, 9, 43, 52, 15, 12, 3, 5, 5, 2, 5, 9, 10, 7, 2, 4, 11, 6, 2, 9, 8, 12, 0]$

ц) $stablo = [-, 95, 52, 43, 29, 23, 28, 15, 12, 17, 8, 15, 9, 19, 7, 8, 3, 5, 2, 5, 9, 10, 7, 2, 4, 11, 6, 2, 9, 8, 12, 0]$

д) $stablo = [-, 95, 43, 52, 29, 23, 28, 15, 8, 7, 19, 9, 15, 8, 17, 12, 3, 5, 2, 5, 9, 10, 7, 2, 4, 11, 6, 2, 9, 8, 12, 0]$

(б) [4 поена] Дат је алгоритам за рачунање збира над неким сегментом почетног низа a. Колико пролазака кроз петљу је потребно да се израчуна збир сегмента чији је опсег $[2, 11]$?

```
// израчунава се збир елемената полазног низа дужине n који се
// налазе на позицијама из сегмента [a, b] на основу сегментног стабла
// које је смештено у низу stablo, кренувши од позиције 1
int saberi(int stablo[], int n, int a, int b) {
    a += n; b += n;
    int zbir = 0;
    while (a <= b) {
        if (a % 2 == 1) zbir += stablo[a++];
        if (b % 2 == 0) zbir += stablo[b--];
        a /= 2;
        b /= 2;
    }
    return zbir;
}
```

Заокружи тачан одговор:

а) 1 б) 2 в) 3 г) 4

Задатак 7. Пројектовање база података

Окидачи у базама података представљају објекте који се активирају аутомацки када се догоди одређени догађај (нпр. INSERT, UPDATE, DELETE) на одређеној табели. Окидачи су механизми који се извршавају након што се догоди акција, али пре него што се промене запишу у бази података. Када се догоди одређени догађај на табели, окидач се активира и покреће. Окидач може референсати на стање табеле пре и после догађаја, те може извршити неку акцију на темељу тог стања, као што је ажурирање друге табеле, унос новог реда у другу табелу или извођење неке друге радње.

Окидаче описују две категорије у које морају да се сврстају:

- тип окидача:
  - BEFORE - извршава се пре операције за коју се окидач везује
  - AFTER - извршава се након операције за коју се окидач везује
- операција за окидање
  - INSERT - окидач се активира приликом уношења новог реда у посматрану табелу
  - UPDATE - окидач се активира приликом ажурирања редова из посматране табеле
  - DELETE - окидач се активира приликом брисања редова посматране табеле

Синтакса писања окидача је следећа: (знак | означава дисјункцију тј. логичко или)

```
create trigger [IME] [before|after] [insert|delete|update] on [IME_TABELE]
 for each row
 begin
      [OPERACIJA_1];
      [OPERACIJA_2];
      ...
      [OPERACIJA_K];
 end $$
 delimiter $$
```

Као што смо напоменули, приликом креирања окидача мора се навести табела за коју се окидач везује, операција коју мотримо (update,insert, delete) као и време окидања (before, after). Пример:

```
create trigger okidac_A AFTER INSERT on tabela_A
for each row
begin
      [operacije];
end
```

Наведени окидач okidac_A би извршавао операције које бисмо навели у његовом телу након уношења новог реда у табелу tabela_A.

Током писања операција у оквиру тела окидача, у зависности од операције за коју је окидач везан можемо референсати на нову (new), стару (old) или обе вредности посматраног атрибута у табели.

Нека је дата база података тест са следећим релацијама:

```
student_mast(STUDENT_ID, NAME, ST_CLASS)
student_marks(STUDENT_ID, NAME,SUB1, SUB2, SUB3, SUB4, SUB5, TOTAL, PER_MARKS)
student_log(USER_ID, DESCRIPTION)
```

(а) [3 поена] Претпоставимо да табела student_marks садржи податке о студентима који су се пријавили за полагање теста. Како тест још није завршен, претпоставимо да се у табели за сваког студента налазе само вредности припадајућих атрибута STUDENT_ID и NAME док су вредности свих осталих атрибута постављене на подразумеване вредности (0 или NULL). Завршен је тест и студенти су добили оцене за 5 различитих предмета. Уносе се оцене у табелу student_marks. Допунити код окидача који аутомацки мења укупну оцену(TOTAL) и просечну оцену (PER_MARKS) свих предмета. Вредности ових атрибута се израчунавају на следећи начин:

укупна оцена: TOTAL = SUB1 + SUB2 + SUB3 + SUB4 + SUB5
просечна оцена: PER_MARKS = (TOTAL)/5

Попунити код одговарајућег окидача:

```
delimiter $
create trigger update_stats 1. _____ 2. _____ on student_marks
for each row
begin
      set new.TOTAL = new.SUB1 + new.SUB2 + new.SUB3 + new.SUB4 + new.SUB5;
      set new.PER_MARKS = new.TOTAL/5;
end $
```

а) 1. before, 2. insert б) 1. after, 2. insert в) 1. before, 2. update г) 1. after, 2. update

(б) [2 поена] Допунити код окидача који приликом промене разреда (st_class) студентима у табели student_mast проверава да унети разред није мањи од досадашњег.

```
delimiter $
create trigger cant_downgrade 1. _____ update on student_mast
for each row
begin
      declare msg varchar(255);
      set msg = 'Greska: Novi razred je manji od starog!';
      if 2. _____.st_class < 3. _____.st_class
          then signal sqlstate '45000' set message_text=msg;
      end if;
end $
```

а) 1. before, 2. old, 3. new б) 1. after, 2. new, 3. old в) 1. before, 2. new, 3. old г) 1. after, 2. old, 3. new

(в) [2 поена] Допунити окидач који обезбеђује да се приликом сваке измене појединачне n-торке у табели student_mast дода нови ред у табели student_log са информацијом о кориснику (user_id) који је извршио измену и описом у вези извршене измене.

```
create trigger log_update 1. _____ 2. _____ on student_mast
for each row
begin
     insert into student_log (USER_ID, DESCRIPTION) values
     (
        user(),
        concat(
           'Izmena podataka o studentu "',
           old.name, ' (', old.student_id, ")", promena: ',
           old.st_class, ' -> ', new.st_class
        )
     );
end $
```

а) 1. before, 2. update б) 1. after, 2. insert в) 1. before, 2. insert г) 1. after, 2. update

(г) [2 поена] Попунити окидач који приликом сваког брисања n-торке из табеле student_mast додаје нови ред у табелу student_log са информацијом о кориснику који врши брисање и кратким описом извршеног брисања.

```
create trigger log_delete 1. _____ delete on student_mast
for each row
begin
    insert into student_log (USER_ID, DESCRIPTION) values
    (
      user(),
      concat(
        'Obrisani su podaci o studentu "',
        2. _____.name, ' (', 3. _____.student_id, ')"'));
end $
```

а) 1. before, 2. new, 3. new б) 1. after, 2. old, 3. old в) 1. before, 2. new, 3. old г) 1. after, 2. new, 3. new

Задатак 8. Функционално програмирање

Функционално програмирање је програмирање које се заснива на појму математичких функција. Основни циљ функционалног програмирања је да опонаша математичке функције - резултат тога је приступ програмирању који је у основи другачији од "класичног", императивног програмирања.

Пример. Претпоставимо да имамо функцију:

$$\max(x, y) = \begin{cases} x, & x > y \\ y, & y \geq x \end{cases}$$

Претходну функцију можемо користити за дефинисање нових функција. На пример:

$$\max3(x, y, z) = \max(\max(x, y), z)$$

Функција max3 је композиција функција max. Претходно дефинисане функције можемо комбиновати на разне начине. На пример, ако треба израчунати $\max6(a, b, c, d, e, f)$, то можемо урадити на следеће начине: $\max(\max3(a, b, c), \max3(d, e, f))$, $\max3(\max(a, b), \max(c, d), \max(e, f))$, $\max(\max(\max(a, b, \max(c, d)), \max(e, f))$. Програмски језик Haskell је пример функционалног програмског језика. Програмирање се заснива на писању функција, попут математичких, и њихове примене. Погледајмо пример Haskell функције:

```
factorial :: Int -> Int
factorial 0 = 1
factorial n = n * factorial (n-1)
```

У овом примеру дефинишемо функцију factorial, која рачуна факторијел броја n. У првој линији дефинишемо пресликавање типова (tipulaz → tipizlaz). У овом случају и улазни и излазни параметар је Int (integer, целобројна вредност). У случају да имамо више улазних вредности, типове наводимо један за другим, а затим на крају наводимо тип излазне вредности (tipulaz1 → tipulaz2 → tipizlaz). Друга линија обрађује случај да је улазни број 0, у том случају је излаз функције 1. Трећа линија обрађује случај када је улаз различит од 0. Тада је излаз једнак ulazniBroj ∗ factorial(ulazniBroj − 1). Овде можемо видети пример рекурзије, тј. функцију која позива саму себе. Рекурзија се веома често виђа у функционалним програмима. Када се функција позове рекурзивно, она ствара нову инстанцу те исте функције која има своје сопствене локалне променљиве и параметре. Свака инстанца функције решава мањи део проблема и враћа резултат. Резултати се затим комбинују да би се добио коначан резултат. Основни типови у програмском језику Haskell су: Bool, Char, Bool, Int, Integer и Float. Веома су сличи као и у "класичним" програмским језицима. Често се користе листе, колекције произвољног броја вредности истог типа (пример: [Int], $[1, 2, 3, 4, 5]$). Листе се могу приказати и у облику (x : xs), где x представља прву вредност у листи, док је xs полазна листа без првог члана. На пример, за листу $[1, 2, 3, 4, 5]$ важи $x = 1$, а $xs = [2, 3, 4, 5]$. Такође, употребљавају се и торке. Торке су колекције фиксираног броја вредности, али потенцијално различитих типова (пример: $(\text{Float}, \text{Int})$). У Haskell-у, симбол | се користи за дефинисање вишеструких грана у функцијама. Вишеструке гране се користе за дефинисање израза који зависе од различитих услова. Последња грана (otherwise) се примењује ако није испуњен ни један од претходно задатих услова.

```
nazivFunkcije :: TipoviArgumenata -> TipRezultata
nazivFunkcije argumenti
  | uslov1 = izraz1
  | uslov2 = izraz2
  | uslov3 = izraz3
  ...
  | otherwise = izrazN
```

Када проверавамо једнакост користимо симбол ==, а када проверавамо неједнакост користимо симбол /=. На пример, вредност $20 == 23$ је False, док је вредност $20/= 23$ једнака True.

(а) [2 поена] Haskell функција је дефинисана на следећи начин (x 'mod' y даје остатак при дељењу броја x бројем y):

```
matematikaFunkcija :: Int -> Int -> [Int] -> [Int]
matematikaFunkcija k n [] = [n]
matematikaFunkcija k n (x:xs)
  | k <= 0 = n : (x:xs)
  | otherwise = (x : (matematikaFunkcija (k - 1) n xs))
```

Шта ће бити резултат позива функције matematikaFunkcija 8 (-1) $[1, 2, 3, 4, 5, 6]$?

а) $[1, 2, 3, 4, 5, 6]$ б) $[-1, 1, 2, 3, 4, 5, 6]$ в) $[1, 2, 3, 4, 5, 6, -1]$ г) $[1, 2, 3, 4, 5, 6, -1, -1]$

(б) [3 поена] Haskell функције су дефинисане на следећи начин (length lst враћа број елемената листе lst):

```
informatikaFunkcija :: Int -> [Int] -> [Int]
informatikaFunkcija k [] = []
informatikaFunkcija k l
  | k <= 0 = l
  | k > length l = l
  | otherwise = astronomijaFunkcija k l

astronomijaFunkcija :: Int -> [Int] -> [Int]
astronomijaFunkcija k [] = []
astronomijaFunkcija k (x:xs)
  | k == 1 = xs
  | otherwise = (x : (astronomijaFunkcija (k - 1) xs))
```

Шта ће бити резултат позива функције informatikaFunkcija 4 $[1, -1, 2, -2, 3, -3]$?

а) $[2, -2, 3, -3, 1, -1]$ б) $[]$ в) $[1, -1, 2, 3, -3]$ г) $[-1, 2, -2, 3, -3]$

(в) [4 поена] Haskell функције су дефинисане на следећи начин:

```
matfFunkcija :: [Double] -> [Double] -> [Double]
matfFunkcija [] p = []
matfFunkcija p [] = []
matfFunkcija (x:xs) ys = dmsFunkcija (sumaFunkcija x ys) (0 :
(matfFunkcija xs ys))

sumaFunkcija :: Double -> [Double] -> [Double]
sumaFunkcija p [] = []
sumaFunkcija c (x:xs) = (c * x : (sumaFunkcija c xs))

dmsFunkcija :: [Double] -> [Double] -> [Double]
dmsFunkcija [] ys = ys
dmsFunkcija xs [] = xs
dmsFunkcija (x:xs) (y:ys) = (x + y : (dmsFunkcija xs ys))
```

Шта ће бити резултат позива функције matfFunkcija $[1.0, 2.0, -1.0, -5.0, 3.0]$ $[3.0, -5.0, 2.0]$?

а) $[3.0, 4.0, -1.0, 9.0, -32.0, 18.0, 6.0]$ б) $[3.0, 1.0, 32.0, -6.0, -11.0, -25.0, 6.0]$

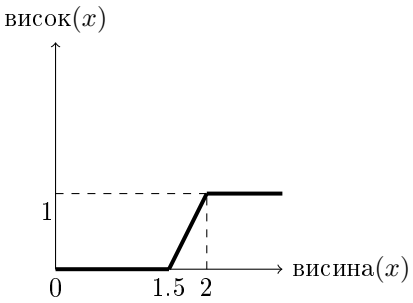
в) $[3.0, -5.0, 2.0, 9.0, -32.0, 22.0, 0.0]$ г) $[3.0, 1.0, -11.0, -6.0, 32.0, -25.0, 6.0]$

Задатак 9. Рачунарска интелигенција

Класична бинарна логика често није применљива у решавању реалних проблема. Ти проблеми се неретко описују недовољно прецизно (нпр. искази "веома гласно" и "помало ветровито" нису бинарни). Зато се јавила потреба за увођењем другачијег логичког апарата који ће обезбедити правила за дефинисање оваквих израза.

Дајемо још један пример како би изложени проблем постао јаснији. Како бисмо дефинисали скуп високих људи? Ако поставимо границу на $175cm$, особе које су високе $176cm$ и $212cm$ биће "једнако високе", док ће особа од $174cm$ бити ниска, иако је само $2cm$ нижа од претходно поменуте високе особе.
Фази скупови нам омогућавају да се припадност елемента неком скупу дефинише нумеричком вредношћу између 0 и 1 (**Пажња!** не треба мешати ове појмове са појмовима из теорије вероватноће иако терминологија у неким ситуацијама може бити слична). Ако је X домен, а $x \in X$ конкретни елемент тог домена, онда се фази скуп A описује функцијом припадности: $\mu_A : X \to [0,1]$.
Фази функција припадности скупу ограничена је између 0 и 1 и за сваки елемент домена је једнозначна. Ево приказа једног могућег начина дефинисања скупа високих људи и фази функције припадности:

$$\text{висок}(x) = \begin{cases} 0, & \text{ако висина}(x) < 1.5m \\ (\text{висина}(x) - 1.5m) * 2.0m, & \text{ако } 1.5m \leq \text{висина}(x) \leq 2.0m \\ 1, & \text{ако висина}(x) > 2.0m \end{cases}$$



Постоји више фази скуповних релација као што су једнакост скупова, подскупови, комплемент, пресек, унија, али овде ћемо приказати само пресек и унију.

Пресек - постоји много начина, а стандардни су:
- преко минимума: $\mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}, \forall x \in X$
- преко производа: $\mu_{A \cap B}(x) = \mu_A(x) \cdot \mu_B(x), \forall x \in X$ (**Пажња!** у овом случају треба бити опрезан, јер већ након неколико изведених множења вредност може тежити нули, у тим ситуацијама је ради лакшег упоређивања боље користити минимум)

Унија - постоји много начина, а стандардни су:
- преко максимума: $\mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}, \forall x \in X$
- преко збира: $\mu_{A \cup B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x), \forall x \in X$ (**Пажња!** у овом случају треба бити опрезан, јер већ након неколико изведених сабирања вредност може тежити јединици, у тим ситуацијама је ради лакшег упоређивања боље користити максимум)

Пример:

$$\mu_{\text{висок}}(\text{петар}) = 0.9 \text{ и } \mu_{\text{атлета}}(\text{петар}) = 0.8$$

$$\mu_{\text{висок}}(\text{марко}) = 0.9 \text{ и } \mu_{\text{атлета}}(\text{марко}) = 0.5$$

Ако се зна да је добар кошаркаш висок и атлета, који од ове двојице је бољи? Применом правила минимума за пресек (логичко и):

$$\mu_{\text{добар кошаркаш}}(\text{петар}) = \min\{0.9, 0.8\} = 0.8$$

$$\mu_{\text{добар кошаркаш}}(\text{марко}) = \min\{0.9, 0.5\} = 0.5$$

па закључујемо да је Петар бољи кошаркаш.

Примена функција припадности над улазним подацима назива се процес фазификације, и тиме смо у горенаведеном примеру добили $\mu_{\text{висок}}(\text{Петар}) = 0.9$, $\mu_{\text{атлета}}(\text{Петар}) = 0.8$ и исто тако за Марка. Затим следи примена правила закључивања. На излазу из правила закључивања је фазификовани излаз за свако од правила (у нашем примеру правило је висок $\wedge$ атлета $\Rightarrow$ добар кошаркаш, а фазификовани излаз: $\mu_{\text{добар кошаркаш}}(\text{Петар}) = \min\{0.9, 0.8\} = 0.8$ и исто тако за Марка).

Шта ако бисмо имали више правила за извођење закључка да је неко добар кошаркаш? Када одредимо вредност за свако правило појединачно, коначну вредност функције припадности рачунамо као унију (максимум) свих добијених вредности. Све ово значи да је на излазу из закључивања степен припадности за сваки од фази скупова закључака (овде је рецимо могло бити и правило када је неко лош или осредњи кошаркаш).

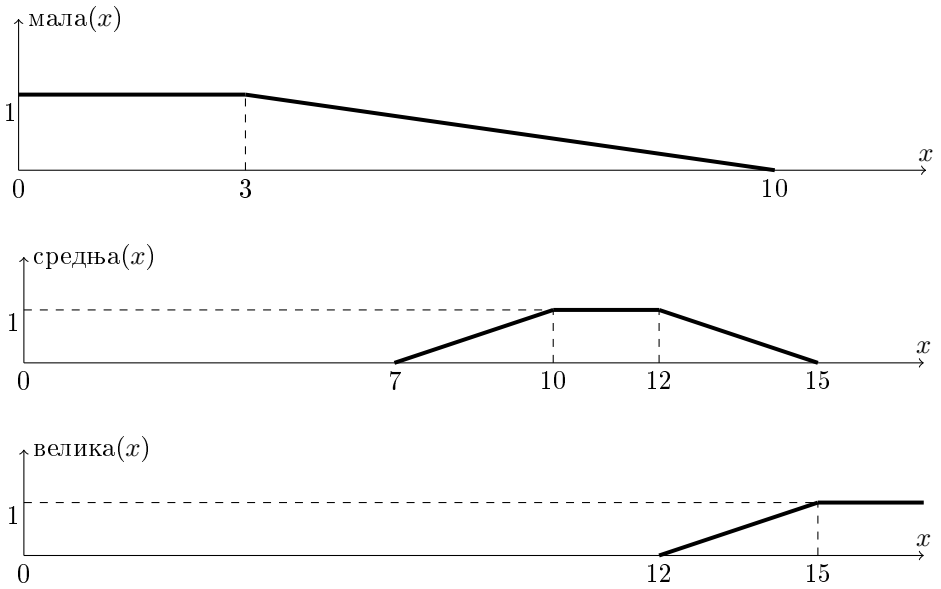
Последњи корак је превођење фази закључака у не-фази (дефазификација). Дефазификацију изводимо по наредној формули:

$$\frac{\sum_{i=1}^{n} c_i * \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)}$$

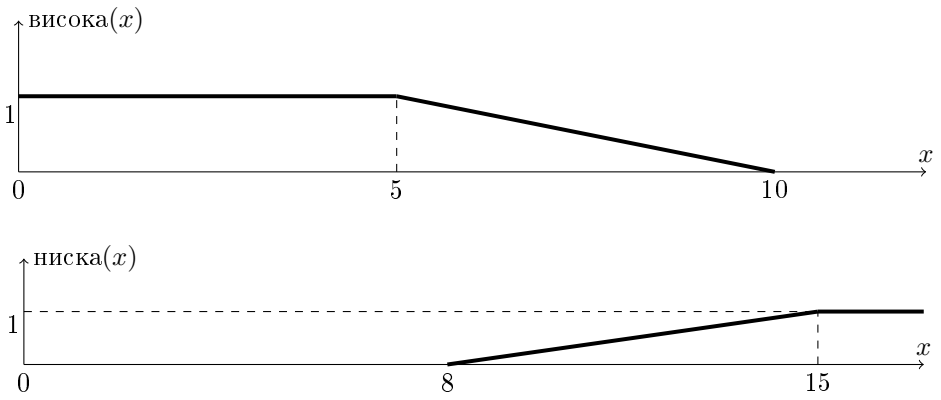
Где $\mu(x_i)$ означава фазификовани излаз и-тог правила (или уније правила ако их има више за исти фази скуп), а c_i константу која ће у задатку бити дата за сваки фази скуп.

У овом задатку одређујемо вредност аутомобила посматрајући његову потрошњу (у литрима горива на $100km$) и поузданост (у просечном броју кварова на $1000km$). Он има потрошњу од 9 $l/100km$ и просечно 8 кварова на $1000km$.

Функције припадности за малу, средњу и велику потрошњу:



Функције припадности за високу и ниску поузданост:



Правила закључивања:

мала потрошња $\wedge$ висока поузданост $\Rightarrow$ велика вредност
мала потрошња $\wedge$ ниска поузданост $\Rightarrow$ средња вредност
средња потрошња $\wedge$ висока поузданост $\Rightarrow$ средња вредност
средња потрошња $\wedge$ ниска поузданост $\Rightarrow$ средња вредност
велика потрошња $\wedge$ висока поузданост $\Rightarrow$ средња вредност
велика потрошња $\wedge$ ниска поузданост $\Rightarrow$ мала вредност

Вредности константе c за:

малу вредност: $c = 7$
средњу вредност: $c = 20$
велику вредност: $c = 40$

(а) [**9 поена**] Колика је приближна вредност аутомобила?

а) 21 б) 23 в) 25 г) 27

Задатак 10. Истраживање података

Истраживање података је процес откривања корисних информација и знања из великих скупова података применом различитих техника и алгоритама. Добијене информације и знања се могу применити на различита подручја као што су маркетинг, медицина, финансије и многа друга.

Једна од техника која се користи у истраживању података јесте класификација. Класификација представља процес категоризације улазних података у различите класе или категорије, на основу њихових особина или карактеристика. Један од примера класификације може бити категоризација имејл порука у spam или ne-spam категорију. У овом случају, улазни подаци су текстуалне поруке е-поште, а циљ је класификовати их у једну од две категорије на основу њихових својства.

Класификација података може бити извршена уз помоћ многобројних алгоритама. Један од таквих јесте алгоритам - kNN (k Nearest Neighbours). Идеја овог алгоритма јесте да посматрамо К најближих суседа податка који желимо да класификујемо и да на основу класа тих суседа закључимо којој класи сам податак припада. Пре него што започнемо класификацију података, морамо одредити следеће параметре. Параметар k представља број најближих суседа податка који класификујемо. Други параметар који посматрамо јесте метрика тј. начин на основу којег рачунамо растојања међу подацима. Параметре који најбоље класификују наше податке одређујемо над подацима за које већ знамо којим класама припадају (такве податке називамо тренинг подацима). Приликом оцењивања колико добро наши параметри класификују податке, можемо на тренутак занемарити којим категоријама сваки податак припада и уз помоћ kNN алгоритма, са одабраним параметрима, одредити нове "предвиђене" класе. Новодобијене класе можемо упоредити са стварним класама наших података и на тај начин оценити за колико података смо тачно предвидели класу којој припадају. Поменути поступак понављамо над различитим комбинацијама параметара и бирамо ону комбинацију која има најбољу оцену тј. најбоље класификује доступне податке. Уз помоћ најбоље оцењених параметара можемо предвиђати класе нових података за које не знамо којој категорији припадају, што и јесте циљ класификације.

У наредном примеру ћемо илустровати поступак класификације kNN алгоритмом. Дати су нам подаци о годинама и месечној заради одређених особа. Циљ нам је да предвидимо да ли је особа купила аутомобил. Ови подаци представљају тренинг податке, и они нам служе, као што смо раније поменули, да оценимо колико добро одабрани параметри класификују доступне податке. Као параметре алгоритма користићемо, на пример, k = 2 и еуклидско растојање.

Године	Месечна зарада	Купио ауто
26	30	Не
31	34	Не
35	43	Да
42	46	Да
48	49	Да

За сваки податак, тј. сваки ред из табеле, посматраћемо којим класама припадају његова два најближа суседа. Податак ћемо класификовати оном класом којој припада већина његових k најближих суседа. Уколико би број суседа који припадају класи "Да" био једнак са бројем суседа који припадају класи "Не", податак можемо класификовати насумично једном класом, у овом примеру ћемо на пример увек класификовати класом "Не". Постоје различите прецизније и сложеније технике које решавају овај случај, али ћемо се због једноставности задржати на овој.

Еуклидско растојање између два податка рачунаћемо помоћу формуле:

$$d((x_1, \ldots, x_n), (x'_1, \ldots, x'_n)) = \sqrt{(x'_1 - x_1)^2 + \ldots + (x_n - x'_n)^2}$$

где n представља број колона које разматрамо, а $x_1, ..., x_n$ и $x'_1, ..., x'_n$ представљају вредности тих колона из табеле. У нашем примеру x и x' се односе на колоне "Године" и "Месечна зарада".

Израчунаћемо растојања првог податка са свим осталим из табеле и његову предвиђену класу ћемо одредити на основу његова 2 најближа суседа. Овај поступак ћемо поновити за сваки податак и на крају упоредити за колико података смо тачно се стварна класа и предвиђена класа поклапају, тј. колико података смо тачно класификовали. Удео тачно класификованих података у односу на укупан број података

представља тачност предвиђања. Израчунајмо растојање између првог и другог податка:

$$d((26,30),(31,34)) = \sqrt{(31-26)^2 + (34-30)^2} = 6.40$$

На исти начин рачунамо и растојања првог податка са осталим подацима, и она износе редом 15, 21.93, 28.42. Други и трећи податак представљају 2 најближа суседа првог податка. Пошто други податак припада класи "Не" а трећи класи "Да", први податак ћемо класификовати класом "Не", по ранијем договору. У случају да су оба податка припадали истој класи, први податак би био класификован том истом класом. Након што смо за сваки податак одредили предвиђену класу, можемо их упоредити са стварним класама и видети колико тачно смо извршили предвиђање. Предвиђене класе наших података биће редом ["НЕ", "НЕ", "НЕ", "ДА", "ДА"]. Пошто су стварне класе података ["НЕ", "'НЕ", "ДА", "ДА", "ДА"], уочићемо да смо тачно класификовали све податке осим трећег, што значи да уз помоћ одабраних параметара, тачност предвиђања класа износи 80%.

Пред тобом се налазе подаци 5 пацијената из једне болнице. Твој задатак је да предвидиш да ли пацијент има ризик од добијања дијабетеса. Подаци који су ти доступни су крвни притисак, индекс телесне масе, количина шећера у крви и индикатор да ли пацијент има дијабетес или не.

Крвни притисак (mmHg)	BMI	Количина шећера (mg/dL)	Дијабетес
1.2	3.4	2.3	Да
2.3	1.4	4.5	Не
3.1	2.2	1.1	Да
4.5	3.2	5.6	Не
2.1	4.1	3.2	Да

Поред еуклидског растојања, још једна метрика помоћу које можемо рачунати растојање међу подацима јесу:

- "Менхетн" растојање и оно се израчунава помоћу следеће формуле:

$$d((x_1,\ldots,x_n),(x'_1,\ldots,x'_n)) = |x_1 - x'_1| + \ldots + |x_n - x'_n|$$

- Чебишевљево растојање

$$d((x_1,\ldots,x_n),(x'_1,\ldots,x'_n)) = \max\{|x_1 - x'_1|,\ldots,|x_n - x'_n|\}$$

(а) [5 поена] Одредити "Менхетн" и Чебишевљево растојање између свака два податка.

(б) [6 поена] Која комбинација параметара најбоље предвиђа класе над датим подацима уколико k бирамо из скупа $\{1,2\}$, а за метрику разматрамо "Менхетн" и Чебишевљево растојање.

(в) [3 поена] Уз помоћ најбоље оцењених параметара под (б), предвидети да ли следећи пацијент има ризик од добијања дијабетеса: {"Количина шећера" = 3.9, "BMI" = 3.6, "Крвни притисак" = 2.8} (*Напомена:* Податак чији једнак број k најближих суседа припада једној, односно другој класи, класификовати класом "Да").