VISOKA POSLOVNA ŠKOLA
STRUKOVNIH STUDIJA
ČAČAK

DIPLOMSKI RAD

Izrada interaktivne animacije u JAVI za simuliranje gibnja tijela u sustavima koji se vrte

Mentor:_____ Student: _____
Profesor: _____ Br.Indeksa: _____

Sadržaj

Uvod

U završnim razredima mog osnovnoškolskog obrazovanja zaključio sam da su fizika i informatika područja koja me zanimaju i u kojima želim napredovati. Išao sam stoga na natjecanja iz fizike, dok sam znanje o informatici nadograđivao programskim jezikom *BASIC*, a potom jezikom *Pascal* u gimnaziji.

Uz daljnje nadograđivanje mog znanja iz fizike i informatike na ovom fakultetu, dobio sam želju da napravim računalnu simulaciju nekog fizikalnog problema, a koji bi mogao poslužiti u nastavi fizike kako bi učenicima, studentima olakšao razumijevanje.

Mentor mi je ponudio temu o gibanju tijela u sustavima koji se vrte, s naglaskom na gibanje tijela pod utjecajem Coriolisove sile. Naime, zamijećeno je da studenti slabije razumijevaju osobito Coriolisovu silu, stoga bi uz nužno izvođenje fizikalnog eksperimenta, u nastavi mehanike bilo poželjno i predstavljanje interaktivne animacije studentima. S tim ciljem, koristit ću programski jezik Java kako bi napravio simulaciju ovog fizikalnog problema.

Ovaj diplomski rad čini prvo poglavlje koje opisuje fizikalni problem - sustav koji se vrti, uz pojavu "prividnih" sila - translacijska, centrifugalna, Coriolisova i azimutalna sila. U drugom poglavlju opisuje se programski jezik Java, izrada simulacije, korištenje simulacije, a potom implementacija simulacije u nastavi fizike.

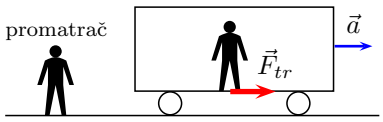
Poglavlje 1

Fizikalni problem

1.1 Inercijski i neinercijski referentni sustavi

Poznato nam je da u inercijskom referentnom sustavu, tj. u sustavu u kojem "slobodno tijelo"[1] miruje ili se giba jednoliko, vrijede Newtonovi zakoni. Međutim, u različitim fizikalnim problemima možemo razmatrati i različite neinercijske (tj. akcelerirane) referentne sustave poput liftova, vrtuljaka i drugih u kojima ne vrijede Newtonovi zakoni. Postavlja se sljedeće pitanje: da li bi mogli kakvim modifikacijama jednadžbi, mogli opisati i objasniti pojave u neinercijskim referentnim sustavima?

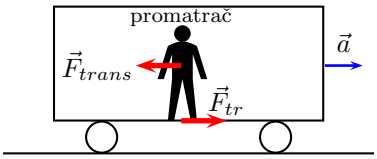
Odgovor na ovo pitanje je "da", uz uvjet da u razmatranje uvedemo neke "prividne" sile (fiktivne sile, inercijske sile ili pseudosile). Te prividne sile su one sile koje osjeća tijelo koje se nalazi u neinercijskom sustavu. Na primjer, zamislimo čovjeka (mase m) koji stoji u vlaku koji ubrzava u desno, akceleracijom $\vec{a}$. Da bi osoba u vlaku ostala na istom mjestu, sila trenja između stopala i poda jednaka je $\vec{F}_{tr} = m \cdot \vec{a}$, usmjerena u desno.



Slika 1.1: Čovjek u vlaku koji ubrzava (promatrajući iz inercijskog sustava - kako uočava promatrač).

U inercijskom sustavu promatrača na Zemlji (slika 1.1), sila trenja $\vec{F}_{tr}$ uzrokuje ubrzanje osobe koja se nalazi u vlaku. U neinercijskom sustavu vlaka (slika 1.2), vrijedi slijedeće: osoba u vlaku osjeća prividnu translacijsku silu $\vec{F}_{trans} = -m \cdot \vec{a}$, usmjerenu u lijevo, a uz navedenu silu trenja $\vec{F}_{tr}$, rezultira mirovanjem osobe u vlaku.

[1] Napomena: "slobodno tijelo" ne međudjeluje sa okolinom.



Slika 1.2: Čovjek u vlaku koji ubrzava (promatrajući u neinercijskom sustavu vlaka).

Ovaj fizikalni problem možemo promatrati i u slučaju kada nema trenja između stopala i poda. Tada u neinercijskom sustavu vlaka djeluje samo translacijska sila $\vec{F}_{trans} = -m \cdot \vec{a}$, što rezultira ubrzanjem osobe prema lijevo, akceleracijom $\vec{a}$. Za isti problem, u inercijskom sustavu promatrača na Zemlji, osoba u vlaku se uopće ne giba.

1.2 Veza koordinata u referentnim sustavima

U prethodnom tekstu uvodi se pojam "prividne" sile uz jednostavni primjer demonstracije (čovjek u vlaku koji ubrzava). Ono što slijedi je pronaći matematički izraz koji će objasniti takve pojave i u drugim neinercijskim referentnim sustavima. Stoga je potrebno naći vezu između inercijskog i neinercijskog koordinatnog sustava u kojima razmatramo fizikalni problem.

Dakle, promotrimo inercijski koordinatni sustav sa osima x_I, y_I, z_I (sa pripadnim jediničnim vektorima $\hat{x}_I$, $\hat{y}_I$, $\hat{z}_I$) i ishodištem O_I, te drugi koordinatni sustav sa osima x, y, z (sa pripadnim jediničnim vektorima $\hat{x}$, $\hat{y}$, $\hat{z}$) i ishodištem O. Kod drugog koordinatnog sustava, osi se mijenjaju na proizvoljan način u odnosu na spomenuti inercijski sustav (tj. ishodište drugog koordinatnog sustava može ubrzavati, a njegove osi mogu rotirati). Vektor $\vec{R}$ je usmjeren od ishodišta O_I do ishodišta O.

Uvodimo česticu čiji je položaj određen vektorima položaja $\vec{r}_I$ u inercijskom sustavu (od O_I do čestice) i vektorom $\vec{r}$ u neinercijskom sustavu (od O do čestice) - promotrimo sliku 1.3. Čestica se giba.

Vrijede slijedeći odnosi:

$$\vec{r}_I = \vec{R} + \vec{r} \tag{1.1}$$

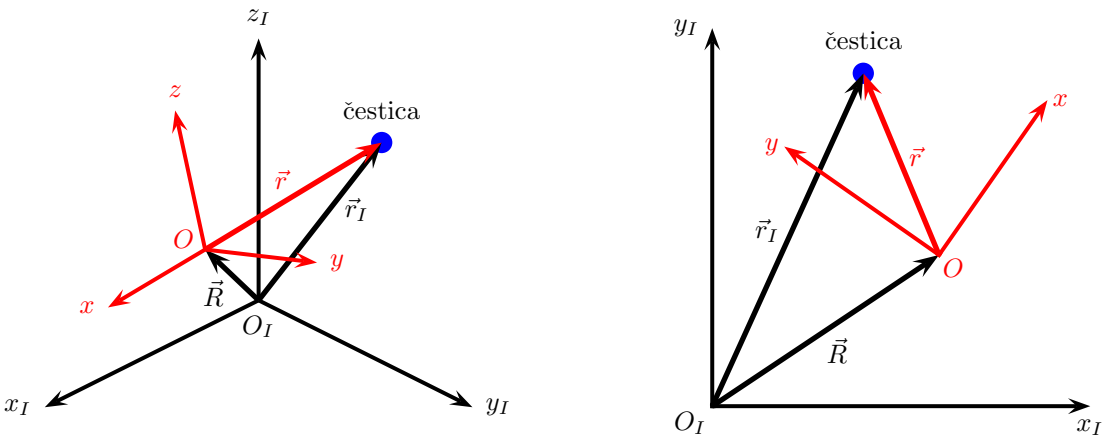
$$\vec{R} = X\hat{x}_I + Y\hat{y}_I + Z\hat{z}_I \tag{1.2}$$

$$\vec{r}_I = x_I\hat{x}_I + y_I\hat{y}_I + z_I\hat{z}_I \tag{1.3}$$

$$\vec{r} = x\hat{x} + y\hat{y} + z\hat{z} \tag{1.4}$$

Izraz (1.1) predstavlja vezu položaja čestice u referentnim sustavima. Potrebno je također pronaći vezu akceleracija čestice u referentnim sustavima (akceleracija čestice je druga derivacija vektora položaja po vremenu).

$$\frac{d^2\vec{r}_I}{dt^2} = \frac{d^2\vec{R}}{dt^2} + \frac{d^2\vec{r}}{dt^2} \tag{1.5}$$



Slika 1.3: Inercijski koordinatni sustav sa osima x_I, y_I, z_I i ishodištem O_I, te neinercijski koordinatni sustav sa osima x, y, z i ishodištem O. Umetnuta čestica se opisuje vektorima položaja $\vec{r}_I$ i $\vec{r}$ u dva spomenuta sustava, koji su povezani sa vektorom $\vec{R}$ (vektorom položaja neinercijskog sustava u odnosu na inercijski sustav). Uz trodimenzionalni prikaz (lijeva slika), radi boljeg razumijevanja koristi se i dvodimenzionalni prikaz, bez z koordinate (desna slika).

Druga derivacija vektora $\vec{r}_I$ ($\frac{d^2\vec{r}_I}{dt^2}$) je akceleracija čestice u inercijskom koordinatnom sustavu - time poznajemo "pravu" silu koja djeluje na česticu: $m\frac{d^2\vec{r}_I}{dt^2} = m\vec{a} = \vec{F}$. Druga derivacija vektora $\vec{R}$ je akceleracija ishodišta drugog, neinercijskog koordinatnog sustava. Preostaje naći drugu derivaciju vektora $\vec{r}$ koja predstavlja akceleraciju čestice u drugom, neinercijskom koordinatnom sustavu.

1.2.1 Izračunavanje $\frac{d^2\vec{r}}{dt^2}$

Prije toga, pronađimo izraz druge derivacije po vremenu nekog općenitog vektora u tom sustavu. U tu svrhu, uvodimo općeniti vektor $\vec{A}$ kojeg predstavljamo preko koordinata neinercijskog koordinatnog sustava.

$$\vec{A} = A_x\hat{x} + A_y\hat{y} + A_z\hat{z}$$

Vremenska derivacija vektora $\vec{A}$:

$$\frac{d\vec{A}}{dt} = \left(\frac{dA_x}{dt}\hat{x} + \frac{dA_y}{dt}\hat{y} + \frac{dA_z}{dt}\hat{z}\right) + \left(A_x\frac{d\hat{x}}{dt} + A_y\frac{d\hat{y}}{dt} + A_z\frac{d\hat{z}}{dt}\right) \tag{1.6}$$

Prvi dio izraza (1.6) je promjena vektora $\vec{A}$ u neinercijskom koordinatnom sustavu, stoga prvi dio izraza (1.6) pišemo kao $\frac{\delta\vec{A}}{\delta t}$. Drugi dio izraza (1.6) se javlja zbog gibajućih koordinatnih osi, tog neinercijskog koordinatnog sustava.

Gibanje takvog sustava je opisano prije navedenim vektorom $\vec{R}$ koji označava gibanje njegovog ishodišta, te rotacijom koordinatnog sustava oko neke osi $\vec{\omega}$ koja prolazi kroz ishodište. Vektor $\vec{\omega}$ je vektor kutne brzine, odnosno vektor kružne

frekvencije koji se može mijenjati u vremenu. Za neki općeniti vektor $\vec{B}$ (konstantnog iznosa) koji rotira kutnom brzinom $\vec{\omega} = \omega\hat{\omega}$ vrijedi[2]:

$$\frac{d\vec{B}}{dt} = \vec{\omega} \times \vec{B}. \tag{1.7}$$

Prema izrazu (1.7) vrijedi $\frac{d\hat{x}}{dt} = \vec{\omega} \times \hat{x}$, odnosno $A_x \frac{d\hat{x}}{dt} = A_x \left(\vec{\omega} \times \hat{x}\right) = \vec{\omega} \times (A_x \hat{x})$. Analogno, dodajući y i z komponentu, preformulirali smo drugi dio izraza (1.6):

$$A_x \frac{d\hat{x}}{dt} + A_y \frac{d\hat{y}}{dt} + A_z \frac{d\hat{z}}{dt} = \vec{\omega} \times (A_x \hat{x} + A_y \hat{y} + A_z \hat{z}) = \vec{\omega} \times \vec{A}.$$

Dakle, vremenska derivacija vektora $\vec{A}$ glasi:

$$\frac{d\vec{A}}{dt} = \frac{\delta \vec{A}}{\delta t} + \vec{\omega} \times \vec{A} \tag{1.8}$$

Vremenska derivacija izraza (1.8):

$$\frac{d^2\vec{A}}{dt^2} = \frac{d}{dt}\left(\frac{\delta \vec{A}}{\delta t}\right) + \frac{d\vec{\omega}}{dt} \times \vec{A} + \vec{\omega} \times \frac{d\vec{A}}{dt} \tag{1.9}$$

Za razvijanje izraza (1.9) koriste se slijedeće transformacije: za prvi član $\frac{d}{dt}\left(\frac{\delta \vec{A}}{\delta t}\right)$ koristi se izraz (1.8), gdje se $\vec{A}$ zamjenjuje sa $\frac{\delta \vec{A}}{\delta t}$. Druga transformacija se tiče trećeg člana, tako da se $\frac{d\vec{A}}{dt}$ zamjenjuje sa izrazom (1.8). Na taj način od izraza (1.9) dobiva se:

$$\frac{d^2\vec{A}}{dt^2} = \left(\frac{\delta^2 \vec{A}}{\delta t^2} + \vec{\omega} \times \frac{\delta \vec{A}}{\delta t}\right) + \left(\frac{d\vec{\omega}}{dt} \times \vec{A}\right) + \vec{\omega} \times \left(\frac{\delta \vec{A}}{\delta t} + \vec{\omega} \times \vec{A}\right)$$

Dakle, tražena druga derivacija po vremenu, općenitog vektora $\vec{A}$ glasi:

$$\frac{d^2\vec{A}}{dt^2} = \frac{\delta^2 \vec{A}}{\delta t^2} + \vec{\omega} \times \left(\vec{\omega} \times \vec{A}\right) + 2\vec{\omega} \times \frac{\delta \vec{A}}{\delta t} + \frac{d\vec{\omega}}{dt} \times \vec{A}. \tag{1.10}$$

Preko dobivenog izraza za općeniti vektor, lako se dobiva analogni izraz za vektor položaja $\vec{r}$. Također, vrijedi da je brzina čestice u neinercijskom sustavu jednaka (brzina izmjerena u tom sustavu):

$$\vec{v} = \frac{\delta \vec{r}}{\delta t}.$$

Druga derivacija vektora $\vec{r}$ po vremenu:

$$\frac{d^2\vec{r}}{dt^2} = \frac{\delta^2 \vec{r}}{\delta t^2} + \vec{\omega} \times (\vec{\omega} \times \vec{r}) + 2\vec{\omega} \times \vec{v} + \frac{d\vec{\omega}}{dt} \times \vec{r}. \tag{1.11}$$

[2]U sfernom sustavu: $\frac{d\vec{r}}{dt} = \frac{dr}{dt}\hat{r} + r\frac{d\theta}{dt}\hat{\theta} + r\sin\theta\frac{d\varphi}{dt}\hat{\varphi}$. U ovom konkretnom slučaju $\frac{dr}{dt} = 0$ i $\frac{d\theta}{dt} = 0$, stoga $\frac{d\vec{r}}{dt} = r\sin\theta\frac{d\varphi}{dt}\hat{\varphi}$. Uz $\omega = \frac{d\varphi}{dt}$, slijedi traženo $\frac{d\vec{r}}{dt} = r\sin\theta\omega\hat{\varphi} = \vec{\omega} \times \vec{r}$.

1.3 Porijeklo "prividnih sila"

Pomoću rezultata iz prethodnog odjeljka, može se utvrditi veza akceleracija čestice u referentnim sustavima u cijelosti. Na taj način, može se dobiti i odnos sila (odnos "pravih" i "prividnih" sila).

Od izraza (1.5) slijedi:

$$\frac{d^2\vec{r}}{dt^2} = \frac{d^2\vec{r}_I}{dt^2} - \frac{d^2\vec{R}}{dt^2}. \tag{1.12}$$

Izjednačujući izraze (1.11) i (1.12), dobiva se:

$$\frac{\delta^2\vec{r}}{\delta t^2} + \vec{\omega} \times (\vec{\omega} \times \vec{r}) + 2\vec{\omega} \times \vec{v} + \frac{d\vec{\omega}}{dt} \times \vec{r} = \frac{d^2\vec{r}_I}{dt^2} - \frac{d^2\vec{R}}{dt^2}.$$

Navedeni izraz pomnoži se sa masom čestice m, tako da $\frac{\delta^2\vec{r}}{\delta t^2}$ stoji na lijevoj strani izraza:

$$m\frac{\delta^2\vec{r}}{\delta t^2} = m\frac{d^2\vec{r}_I}{dt^2} - m\frac{d^2\vec{R}}{dt^2} - m\vec{\omega} \times (\vec{\omega} \times \vec{r}) - 2m\vec{\omega} \times \vec{v} - m\frac{d\vec{\omega}}{dt} \times \vec{r}.$$

Kao što je spomenuto u odjeljku 1.2, član $m\frac{d^2\vec{r}_I}{dt^2} = m\vec{a} = \vec{F}$ predstavlja "pravu" silu koja djeluje na česticu - može biti težna sila, sila trenja itd. U prethodni izraz, uvodimo sile ($\vec{F}_{trans}$ - translacijska sila, $\vec{F}_{cf}$ - centrifugalna sila, $\vec{F}_{cor}$ - Coriolisova sila, $\vec{F}_{azim}$ - azimutalna sila):

$$m\frac{\delta^2\vec{r}}{\delta t^2} = \vec{F} + \vec{F}_{trans} + \vec{F}_{cf} + \vec{F}_{cor} + \vec{F}_{azim}. \tag{1.13}$$

"Prividne" sile:

$$\vec{F}_{trans} = -m\frac{d^2\vec{R}}{dt^2} \qquad \vec{F}_{cf} = -m\vec{\omega} \times (\vec{\omega} \times \vec{r})$$

$$\vec{F}_{cor} = -2m\vec{\omega} \times \vec{v} \qquad \vec{F}_{azim} = -m\frac{d\vec{\omega}}{dt} \times \vec{r}$$

Na tijelo u takvom sustavu djeluju ove spomenute sile. Lijeva strana izraza (1.13) je umnožak mase i akceleracije koja je izmjerena u neinercijskom sustavu. Kako bi osoba u tom sustavu mogla izračunati lijevu stranu izraza (1.13), u obzir treba uzeti "pravu" silu $\vec{F}$ i ostale "prividne" sile u formi $\vec{F} = m\vec{a}$:

$$m\frac{\delta^2\vec{r}}{\delta t^2} = m\vec{a}_{izm} = \sum_i \vec{F}_i.$$

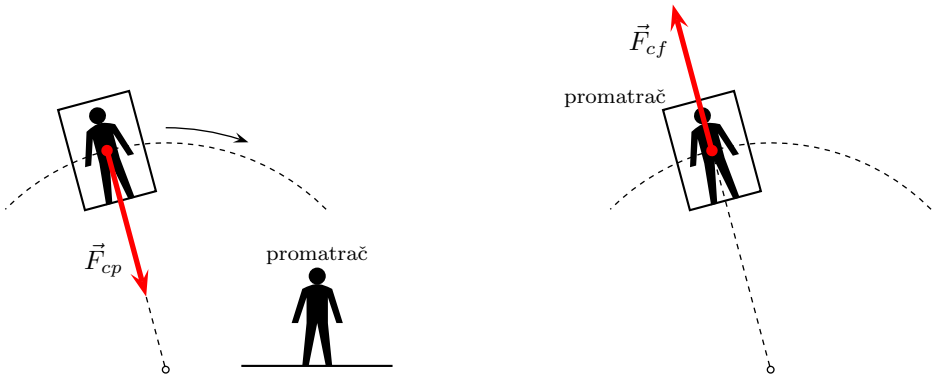
U ovom izrazu veličina $\sum_i \vec{F}_i$ predstavlja vektorski zbroj svih sila, pravih i prividnih, koje osjeća tijelo u neinercijskom sustavu.

1.3.1 Translacijska sila - $\vec{F}_{trans}$

Ovo je najintuitivnija prividna sila. Primjer je naveden u prvom odjeljku (slika 1.2), u situaciji kada se čovjek nalazi u vlaku koji ubrzava. Ako vektor $\vec{R}$ govori o poziciji vlaka (druga derivacija po vremenu je akceleracija vlaka), javlja se prividna translacijska sila $\vec{F}_{trans} = -m\frac{d^2\vec{R}}{dt^2}$ koju osjeća osoba koja se nalazi u vlaku koji ubrzava (odnosno usporava).

1.3.2 Centrifugalna sila - $\vec{F}_{cf}$

Centrifugalna sila na neko tijelo se javlja u neinercijskom sustavu koji se vrti sa određenim vektorom kutne brzine $\vec{\omega}$ (taj vektor određuje os i smjer vrtnje, a iznos brzinu vrtnje). Za isti slučaj, gledajući u inercijskom sustavu, na tijelo koje se kružno giba djeluje centripetalna sila $\vec{F}_{cp}$ (slika 1.4).



promatrajući iz inercijskog sustava — promatrajući iz neinercijskog sustava

Slika 1.4: Čovjek u kabini koja rotira, promatrano u inercijskom sustavu (centripetalna sila) i neinercijskom sustavu rotirajuće kabine (centrifugalna sila).

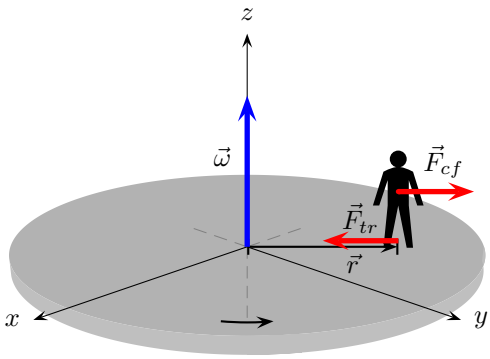
Ovdje je važno naglasiti da centrifugalna sila djeluje u neinercijskom sustavu koji se vrti, dok centripetalna sila djeluje u inercijskom sustavu. Premda su iznosi ovih sila jednaki, ove sile se ne poništavaju jer djeluju u različitim sustavima.

Primjer - Čovjek koji stoji na vrtuljku

Dakle, osoba mase m stoji na vrtuljku na udaljenosti r od središta vrtnje (slika 1.5). Vrtuljak rotira u x-y ravnini sa kutnom brzinom $\vec{\omega} = \omega \cdot \hat{z}$. Vektor $\vec{r} = r \cdot \hat{r}$ leži na x-y ravnini, a polazi od središta vrtnje do položaja osobe na vrtuljku. Postavlja se pitanje: "Kakva je centrifugalna sila u ovom slučaju?".

Prema izrazu za centrifugalnu silu vrijedi:

$$\vec{F}_{cf} = -m\vec{\omega} \times (\vec{\omega} \times \vec{r}) = m\omega^2 r \cdot \hat{r}.$$

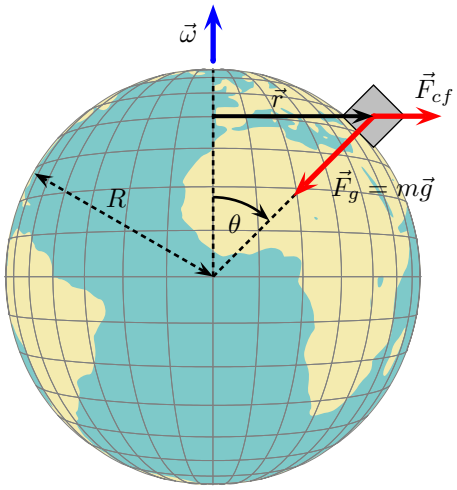


Slika 1.5: Čovjek mase m stoji na vrtuljku koji rotira (promatrajući u neinercijskom sustavu vrtuljka).

Kao što dobiveni izraz govori, centrifugalna sila je usmjerena radijalno od osi vrtnje prema van. Kako bi čovjek mirovao u sustavu vrtuljka (u neinercijskom sustavu), djeluje sila trenja sa istim iznosom kao centrifugalna sila, ali suprotnog smjera. Gledajući ovu situaciju iz inercijskog sustava, na tijelo djeluje centripetalna sila zbog navedene sile trenja.

Primjer - Efektivna težna sila

Promotrimo tijelo koje mirno stoji na zemlji (slika 1.6), na mjestu određenim polarnim kutem kutem θ (kut u odnosu na sjeverni pol). Problem se razmatra u rotirajućem sustavu zemlje.



Slika 1.6: Na osi rotacije Zemlje leži sjeverni i južni pol, s time da je rotacija Zemlje određena kutnom brzinom $\vec{\omega}$. Vektor položaja $\vec{r}$ je usmjeren od osi rotacije do mjesta gdje se tijelo nalazi ($\vec{r}$ je okomit na $\vec{\omega}$). Polumjer Zemlje je R.

Kao i u prethodnom primjeru, koristi se vektor kutne brzine $\vec{\omega}$ i vektor položaja

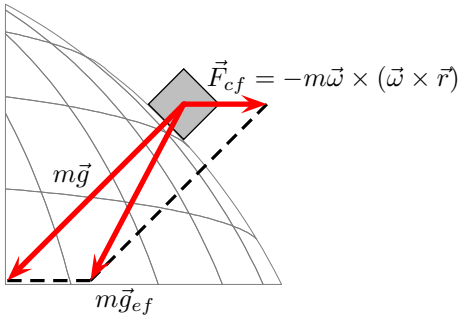
$\vec{r}$. Prvo, izračunajmo kutnu brzinu Zemljine rotacije ω:

$$\omega = \left(\frac{2\pi\ rad}{24 \cdot 3600\ s}\right)\left(\frac{366,5}{365,5}\right) = 7,29 \cdot 10^{-5}\ rad/s.$$

U ovom izrazu, prva zagrada je odnos punog okreta ($2\pi\ rad = 360°$) i vremena koje je potrebno za puni okret. Druga zagrada predstavlja ispravak, tj. odnos sideričkih i sunčevih dana.

Na tijelo djeluje težna sila $\vec{F}_g$, koja je usmjerena radijalno prema centru Zemlje (slika 1.6). Budući da Zemlja rotira oko svoje osi (sustav Zemlje je neinercijski sustav), na tijelo djeluje i centrifugalna sila koja je usmjerena okomito od osi rotacije, prema van (slika 1.6). Zbroj težne i centrifugalne sile daje efektivnu težnu silu (slika 1.7) koja djeluje na tijelo:

$$m\vec{g}_{ef} = \vec{F}_g + \vec{F}_{cf} = m\left[\vec{g} - \vec{\omega} \times (\vec{\omega} \times \vec{r})\right].$$



Slika 1.7: Vektorski zbroj težne i centrifugalne sile.

Centrifugalna sila se u ovom konkretnom primjeru može zapisati preko poznatog polumjera Zemlje R i navedenog kuta θ:

$$\vec{F}_{cf} = -m\vec{\omega} \times (\vec{\omega} \times \vec{r}) = m\omega^2 R \sin\theta \cdot \hat{r}.$$

Po iznosu, ova sila tvori najviše 0,3% težne sile (na ekvatoru), stoga sustav Zemlje u mnogim problemima možemo uzeti kao inercijski sustav.

Zbog ovisnosti o kutu θ, centrifugalna sila je najveća na ekvatoru, a najmanja na polovima. Isaac Newton je promatrajući Jupiter utvrdio da je taj planet u ekvatoru malo izdužen, a na polovima spljošten. Newton je taj učinak objasnio centrifugalnom silom koja se javlja zbog rotacije planeta. Taj efekt je prisutan i na Zemlji - promjer Zemlje od pola do pola je otprilike 42 kilometara manji od promjera Zemlje u ekvatorskoj ravnini.

Rezultat Zemljine spljoštenosti i utjecaja centrifugalne sile očituje se u različitim vrijednostima efektivne akceleracije slobodnog pada $\vec{g}_{ef}$ izmjerene u različitim mjestima na Zemlji (tablica 1.1).

Mjesto	Zemljopisna širina	Nadmorska visina $[m]$	$\vec{g}_{ef}$ $[m/s^2]$
Sjeverni pol	90° Sj.	0	9,832
Grenland	70° Sj.	70	9,825
Stockholm	59° Sj.	45	9,818
Zagreb	46° Sj.	135	9,807
Denver	40° Sj.	1638	9,796
San Francisco	38° Sj.	114	9,800
Panamski kanal	9° Sj.	6	9,782
Novi Zeland	37° Juž.	3	9,800

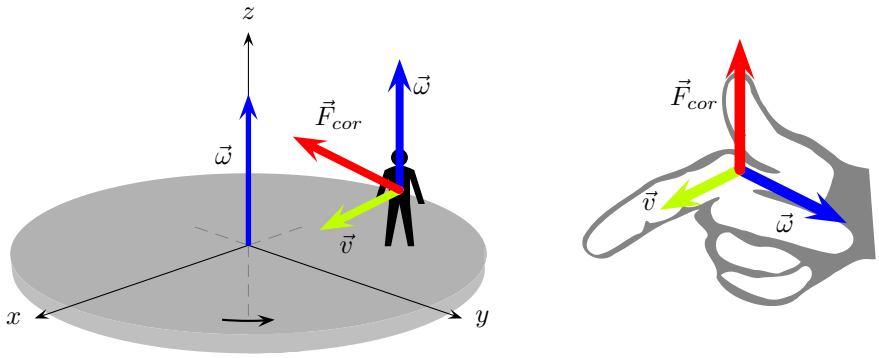
Tablica 1.1: Izmjerene vrijednosti akceleracije slobodnog pada

1.3.3 Coriolisova sila - $\vec{F}_{cor}$

U neinercijskom sustavu koji rotira (koji se vrti) preko određenog vektora kutne brzine $\vec{\omega}$, osim centrifugalne sile, na tijelo djeluje i Coriolisova sila ukoliko se tijelo giba u tom neinercijskom sustavu (tijelo se giba brzinom $\vec{v}$). Coriolisova sila je okomita na vektor kutne brzine $\vec{\omega}$ i vektor brzine $\vec{v}$ kojom se tijelo giba u tom sustavu, na što upućuje izraz za Coriolisovu silu:

$$\vec{F}_{cor} = -2m\vec{\omega} \times \vec{v} = 2m\vec{v} \times \vec{\omega}.$$

Promotrimo jednostavan primjer (slika 1.8) - tijelo mase m na vrtuljku koji rotira kutnom brzinom $\vec{\omega}$. U sustavu vrtuljka, tijelo u početnom trenutku miruje, pa na njega djeluje centrifugalna sila (usmjerena radijalno od osi vrtnje prema van) i sila trenja (u suprotnom smjeru). Ukoliko je centrifugalna sila veća od sile trenja, tijelo se počinje gibati radijalno od osi vrtnje.



Slika 1.8: Čovjek na vrtuljku koji rotira. Prema izrazu za Coriolisovu silu, vektor brzine $\vec{v}$ i vektor kutne brzine $\vec{\omega}$ su okomiti na spomenutu silu (pravilo desne ruke). U ovom konkretnom slučaju, vektori $\vec{v}$ i $\vec{\omega}$ su također okomiti. Radi boljeg razumijevanja, nije prikazana centrifugalna sila, kao ni pripadne sile trenja.

Budući da se tijelo giba, na njega djeluje i Coriolisova sila. Na tijelo pritom djeluje i sila trenja koja je suprotna smjeru Coriolisove sile (u ovom slučaju Cori-

olisova sila je veća od spomenute sile trenja). Kao rezultat, putanja tijela se otklanja u stranu (ovisno o smjeru rotiranja vrtuljka), ako se promatra u sustavu vrtuljka.

Primjer - Ispuštena lopta

Na Zemlji, pri polarnom kutu θ, sa visine h od zemljine površine pušta se lopta. Zemljina rotacija je određena vektorom kutne brzine $\vec{\omega}$. Kako Coriolisova sila utječe na putanju lopte?

Lopta se giba brzinom $\vec{v}$ u nekom trenutku, radijalno prema središtu Zemlje. Veza između brzine, vremena, akceleracije slobodnog pada i visine:

$$v = gt, \qquad h = \frac{gt^2}{2}.$$

Zbog gibanja tijela, promatrajući sa Zemlje, javlja se Coriolisova sila koja je usmjerena istočno (neovisno o Zemljinoj polutki) sa iznosom $2m\omega v \sin\theta$. Očigledno, lopta se time otklanja istočno. Za konkretan primjer[3] (zanemarujući otpor zraka), uz Zemljinu kutnu brzinu $\omega \approx 7,3 \cdot 10^{-5} rad/s$, te npr. kut $\theta = \frac{\pi}{2}$ i visinu $h = 100\ m$, otklon prema istoku iznosi $d_{ist.} \approx 2\ cm$.

Primjer - Foucaultovo njihalo

Foucaultovo njihalo je klasični primjer posljedice Coriolisove sile, a ujedno i jasan dokaz rotacije Zemlje. Rezultat Foucaultovog njihala je da zbog rotacije Zemlje, ravnina njihala rotira sa određenom, mjerljivom frekvencijom, odnosno kutnom brzinom (kako uočava promatrač na Zemlji).

U razmatranju se pretpostavlja da na ovjes njihala ne djeluje trenje, tako da nema momenta sile koji bi zakrenulo ravninu njihala.

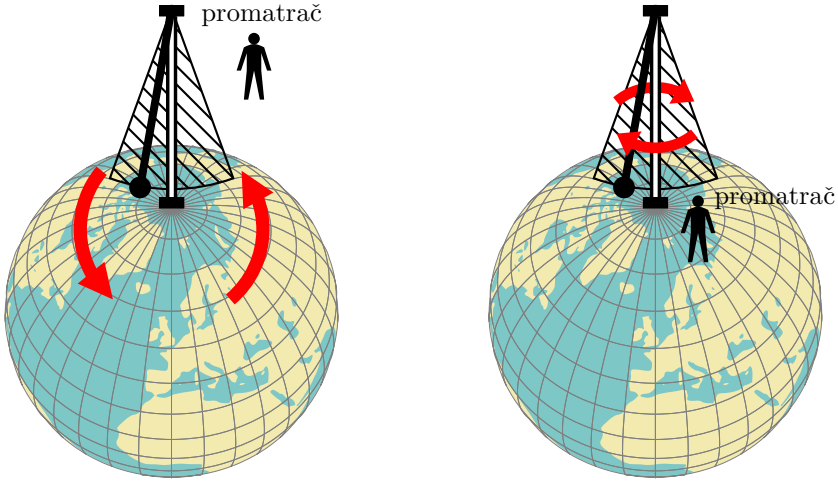
Promatra se slučaj kada se njihalo nalazi na sjevernom polu, u kojem je jednostavno razumjeti rotaciju njihala. Promatrač koji "lebdi" iznad sjevernog pola uočava da Zemlja rotira u smjeru suprotnom kazaljki na satu, dok je ravnina njihala stalna, ona ne rotira (slika 1.9).

S druge strane, promatrač koji mirno stoji na Zemlji uočava kako ravnina njihala rotira u smjeru kazaljke na satu (slika 1.9), tako da ravnina njihala načini jedan okret svaki dan. Kutna brzina ove rotacije je dakako i kutna brzina Zemljine rotacije.

Slijedi razmatranje slučaja kada se njihalo[4] ne nalazi na polovima (već između

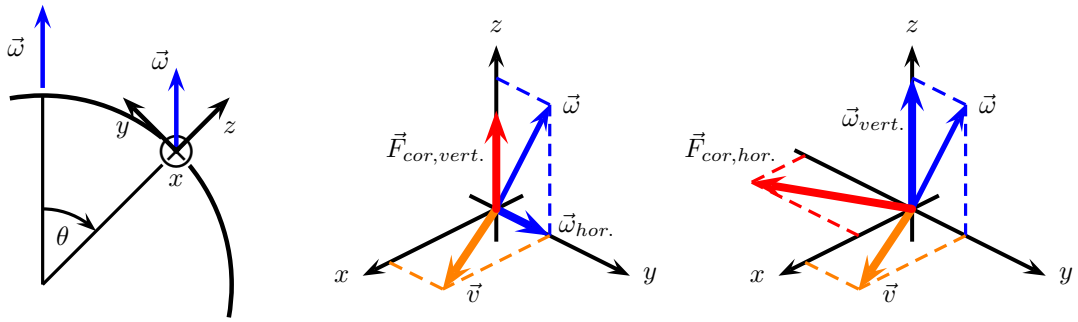
[3]Kako se lopta otklanja istočno, lopta ima tangencijalnu komponentu brzine. Ta komponenta brzine uzrokuje Coriolisovu silu drugog reda, no u ovom problemu možemo zanemariti ovaj mali efekt.

[4]U tu svrhu, koristiti će se aproksimacija da se kuglica njihala giba horizontalno (u odnosu na površinu Zemlje). Ako je nit njihala dugačka, ova aproksimacija je uglavnom zadovoljena. Razlog ovoj aproksimaciji je ispravak zbog dizanja i spuštanja kuglice njihala, dakle taj efekt se aproksimacijom zanemaruje.



Slika 1.9: Promatranje Foucaultovog njihala u sustavu promatrača koji lebdi iznad Zemlje (slika lijevo), te u sustavu Zemlje (slika desno). Promatrač koji lebdi iznad Zemlje uočava rotaciju Zemlje, dok je ravnina njihala stalna. Za promatrača na Zemlji, rotira samo ravnina njihala.

polova), u mjestu određenom polarnim kutem θ (slika 1.10 a)). I u ovom slučaju, važno je doći do kutne brzine rotiranja ravnine njihala. Radi lakšeg razmatranja, na mjestu gdje se nalazi njihalo, definira se koordinatni sustav takav da su njegove osi x i y tangente (kuglica njihala se giba po x-y ravnini, dakle brzina $\vec{v}$ leži u toj ravnini), a z-os normala na površinu Zemlje.



a) b) c)

Slika 1.10: U lokalnom koordinatnom sustavu na Zemlji (a), vektor kutne brzine se razdvaja u horizontalnu (b) i vertikalnu komponentu (c). Budući da brzina leži u x-y ravnini, djeluju komponente Coriolisove sile.

Kutna brzina Zemljine rotacije $\vec{\omega}$ i Coriolisova sila $\vec{F}_{cor}$ se u navedenom koordinatnom sustavu mogu podijeliti u horizontalnu (u x-y ravnini) i vertikalnu komponentu (z-os) - (slika 1.10 b) i c)).

$$\vec{\omega} = \omega \sin\theta\, \hat{y} + \omega \cos\theta\, \hat{z} = \omega_{hor.}\, \hat{y} + \omega_{vert.}\, \hat{z} = \vec{\omega}_{hor.} + \vec{\omega}_{vert.}$$

$$\vec{F}_{cor} = \vec{F}_{cor,\,hor.} + \vec{F}_{cor,\,vert.}$$

Za daljnji razvoj problema relevantna je komponenta Coriolisove sile koja leži u horizontalnoj ravnini $\vec{F}_{cor,\,hor.}$, budući da vertikalna komponenta $\vec{F}_{cor,\,vert.}$ neznatno mijenja težnu silu (ta komponenta je mnogo manja od težne sile, pa nije relevantna za daljnje razmatranje).

Za traženu horizontalnu komponentu Coriolisove sile odgovorna je vertikalna komponenta kutne brzine Zemljine rotacije $\vec{\omega}_{vert.}$ (također, vrijedi i obrnuto).

$$\vec{F}_{cor} = -2m\vec{\omega} \times \vec{v} \quad \Longrightarrow$$

$$\vec{F}_{cor,\,hor.} = -2m\vec{\omega}_{vert.} \times \vec{v} \qquad \vec{F}_{cor,\,vert.} = -2m\vec{\omega}_{hor.} \times \vec{v}$$

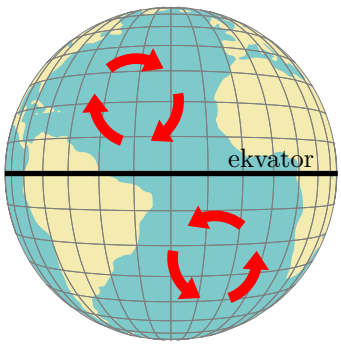
$$\vec{F}_{cor,\,hor.} = -2m\omega \cos\theta\, \hat{z} \times \vec{v}$$

Dakle, u x-y ravnini, na kuglicu njihala koja se giba brzinom $\vec{v}$, djeluje horizontalna komponenta Coriolisove sile $\vec{F}_{cor,\,hor.}$, što rezultira rotiranjem ravnine Foucaultovog njihala sa kutnom brzinom $\omega \cos\theta$. Na sjevernoj polutki ravnina Foucaultovog njihala rotira u smjeru kazaljke na satu, a na južnoj polutki suprotno.

Ostali primjeri

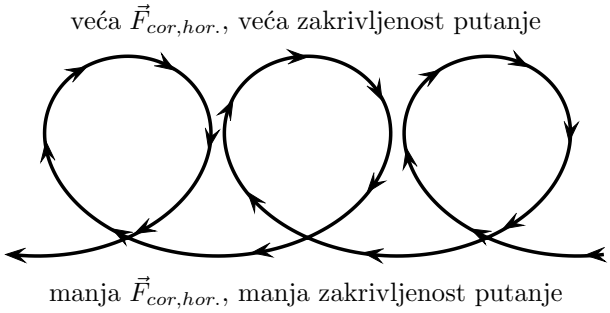
Općenito, Coriolisova sila na sjevernoj polutki otklanja tijela koja se gibaju po površini prema desno, a na južnoj polutki prema lijevo.

Kao rezultat, na sjevernoj polutki anticiklone (područja visokog tlaka) i morske struje se kreću u smjeru kazaljke na satu (slika 1.11), dok se ciklone (područja niskog tlaka) kreću suprotno kazaljki na satu (zbog prevladavajućeg gradijenta tlaka). Na južnoj polutki vrijedi obrnuto.



Slika 1.11: Smjer gibanja anticiklona i morskih struja.

Drugi primjer - gibajuća plutača (bova) na moru zbog Coriolisove sile (i ovisnosti sile o geografskoj širini) opisuje spiralnu putanju (slika 1.12). Ukoliko bi se problem ograničio na manje područje, gdje je Coriolisova sila praktički jednaka, tada bi tijelo imalo kružnu putanju.



Slika 1.12: Približno gibanje plutače na sjevernoj zemljinoj polutki. Putanja plutače je spiralna budući da je horizontalna komponenta Coriolisove sile (slika 1.10) veća ukoliko se tijelo nalazi bliže sjevernom ili južnom polu. Rezultat je da uz približno istu brzinu gibanja, putanja više zakrivljena kada se tijelo nalazi bliže polu, odnosno manje zakrivljena kada se tijelo nalazi bliže ekvatoru.

Umjetna gravitacija - svemirski kotač

U mogućim dugotrajnim svemirskim putovanjima za astronaute je važna umjetna gravitacija, budući da dugotrajan boravak u beztežinskom stanju nepovoljno utječe na zdravlje čovjeka.

Kao jedno od rješenja, nameće se rotirajući svemirski kotač u kojem prisutna centrifugalna sila daje umjetnu gravitaciju. Međutim, u svemirskom kotaču Coriolisova sila je otprilike 4000 puta veća nego na Zemlji. U takvom slučaju stroj sa pokretnim ili rotirajućim dijelovima bi se vjerojatno uništio, dok bi na posadu djelovala fiziološki i psihološki neudobna Coriolisova sila.

Razmatrani sustav	Kutna brzina $\omega\ [\frac{rad}{s}]$	$\frac{F_{cf}}{m}\ [\frac{m}{s^2}]$	$\frac{F_{cor}}{m}\ [\frac{m}{s^2}]$	
			$v_1 = 1\,\frac{m}{s}$	$v_2 = 10\,\frac{m}{s}$
Zemlja	$7,29 \cdot 10^{-5}$	$0,03$	$1,46 \cdot 10^{-4}$	$1,46 \cdot 10^{-3}$
Svemirski kotač	$0,31$	$9,81$	$0,63$	$6,28$

Tablica 1.2: Usporedba relevantnih veličina na Zemlji i rotirajućem svemirskom kotaču polumjera $r = 100\,m$. Zadnja tri stupca daju odnos centrifugalne i Coriolisove sile, odnosno pripadnih akceleracija. U razmatranju se uzima da je brzina okomita na kutnu brzinu ($\vec{v} \perp \vec{\omega}$), osim toga, razmatra se gibanje manjom brzinom v_1, a potom većom brzinom v_2.

1.3.4 Azimutalna sila - $\vec{F}_{azim}$

Ova pseudosila za razliku od ostalih, zahtjeva promjenu vektora kutne brzine $\vec{\omega}$ u vremenu, kao što upućuje izraz za azimutalnu silu:

$$\vec{F}_{azim} = -m\frac{d\vec{\omega}}{dt} \times \vec{r}.$$

Vektor kutne brzine čini iznos i smjer, koji se dakle u vremenu mogu mijenjati:

$$\vec{\omega} = \omega\,\hat{\omega} \quad \Longrightarrow \quad \frac{d\vec{\omega}}{dt} = \frac{d\omega}{dt}\,\hat{\omega} + \omega\,\frac{d\hat{\omega}}{dt}$$

U primjeru ograničiti ćemo se na slučaj kada se vektor kutne brzine mijenja samo u iznosu ($\frac{d\omega}{dt} \neq 0$, $\frac{d\hat{\omega}}{dt} = 0$). U takvom slučaju, azimutalna sila se može zapisati kao:

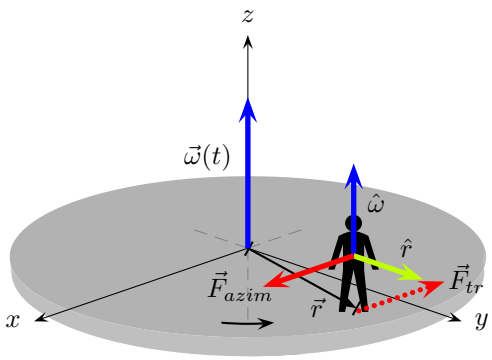
$$\vec{F}_{azim} = -m\frac{d\omega}{dt}\,\hat{\omega} \times \vec{r}.$$

Primjer - Čovjek na vrtuljku koji ubrzava

Čovjek mase m stoji na vrtuljku na udaljenosti r od središta vrtnje. Vrtuljak rotira sa povećanjem iznosa kutne brzine.

Promatrajući iz inercijskog laboratorijskog sustava, na osobu djeluje tangencijalna sila trenja[5] koja je jednaka $ma_{tang.}$, gdje je $a_{tang.} = r\frac{d\omega}{dt}$ tangencijalna akceleracija izmjerena u laboratorijskom sustavu. Ako se problem razmatra u rotirajućem neinercijskom sustavu gdje čovjek miruje, javlja se azimutalna sila koja u tom sustavu izjednačava tangencijalnu silu trenja.

$$\vec{F}_{azim} = -m\frac{d\omega}{dt}\,\hat{\omega} \times \vec{r} = -mr\frac{d\omega}{dt}\,\hat{\omega} \times \hat{r} \qquad \left|\vec{F}_{azim}\right| = mr\frac{d\omega}{dt}$$



Slika 1.13: Čovjek na vrtuljku koji ubrzano rotira. Radi boljeg razumijevanja, nije prikazana centrifugalna sila, kao ni pripadna sile trenja.

[5]Dakako, djeluje i radijalna sila trenja (prema središtu vrtnje) - pa na osobu djeluje centripetalna sila.

Poglavlje 2

Simulacija problema pomoću programskog jezika Java

2.1 Zašto Java?

U poznatim programskim jezicima C, Fortran, Pascal... izvorni kôd programa se temelji na nekom algoritmu, tj. izvorni kôd je niz instrukcija, funkcijskih poziva. No s razvojem računalne tehnologije, noviji programi su sve složeniji, a time i izvorni kôd programa u navedenim jezicima je sve kompleksniji i teži za izmjene, dodavanja, te razumijevanje u cjelini.

Kako bi se riješio, odnosno ublažio ovaj problem, stvoren je novi pristup u programiranju - objektno orijentirano programiranje - uz ključne pojmove: objekt, klasa, podatak i metoda.

Ovaj pristup je ustvari preslika čovjekove apstrakcije, shvaćanja i logike koje su objektno orijentirane. Na primjer, za konkretnog psa (koji predstavlja "objekt", dok općenito pas predstavlja "klasu") vrijede "podaci": ime, boja, masa, pasmina... i "metode": jesti, spavati, lajati, mahati repom...

Prema ovom i sličnim primjerima iz stvarnog svijeta, analogno vrijedi u objektno orijentiranom programiranju:

- Izvorni kôd programa se temelji na skupu objekata koji međusobno komuniciraju, dok je značenje algoritma drugorazredno.
- Objekt pritom sadrži podatke (koji predstavljaju stanje objekta) i metode (tj. funkcije koje služe za promjenu stanja objekta, te za komunikaciju sa drugim objektima).
- Objekt je instanca neke klase, odnosno klasa predstavlja tip objekta, pa prema tome izvorni kôd programa čine klase.

Od više postojećih objektno orijentiranih programskih jezika, zbog određenih pogodnosti (odjeljak 2.2.1), odabrat ćemo Javu.

2.2 Osnovno o Javi

Java danas podrazumijeva mnoštvo tehnologija, alata, proizvoda tvrtke "Sun Microsystems", među kojima spada: Java programski jezik, Java platforme, Java virtualni stroj (eng. *Java Virtual Machine - JVM*), Java izvedbena okolina (eng. *Java Runtime Environment - JRE*), Java razvojna oprema (eng. *Java Development Kit - JDK*) i dr.

Razvoj Jave započinje krajem 1990. godine u tvrtki "Sun Microsystems" pod nazivom "Oak", kao alternativa programskim jezicima C, odnosno C++. Pod vodstvom Jamesa Goslinga, sredinom 1990-ih, taj projekt dobiva novi naziv Java, a prvo javno pojavljivanje događa se sredinom 1995. godine.

2.2.1 Značajke Jave kao programskog jezika

- **Sličnost u odnosu na C++, jednostavnost.** Nema potrebe za pokazivačima, datotekama zaglavlja, opterećenje operatora, itd. Programiranje u Javi uvelike olakšava prisustvo biblioteke programskih elemenata (eng. *Java Application Interface - Java API*) koja sadrži skup gotovih softverskih komponenata.
- **Neovisnost o hardveru i operacijskom sustavu.** Vidjeti odjeljak (2.2.4).
- **Java programski jezik je objektno orijentirani jezik.**
- **Robusnost i sigurnost.** Pouzdano izvršavanje programa, provjera kôda od grješaka pri prevođenju i izvršavanju. Automatsko oslobađanje memorije (za razliku od Jave, programer u C++ mora sam osloboditi korištenu memoriju). Zaštita od virusa i ostalih zloćudnih programa.
- **Višedretvenost** (eng. *multithreading*). Omogućava paralelno (istovremeno) ili prividno paralelno izvršavanje više cjelina (dretvi) nekog programa.
- **Dinamičnost.** Dinamično povezivanje resursa (u C++ vrijedi statičko povezivanje resursa koje može izazvati grješke).
- **Distribuiranost.** Prisutno je mnoštvo mrežnih mogućnosti u okviru Javine biblioteke programskih elemenata (Java API).

Primjena Jave u računalnom svijetu je danas široka, no Java se ponajviše koristi kod mobilnih uređaja, te za poslovne i internet aplikacije.

2.2.2 Java platforma

Spomenuta Java platforma se po namjeni dijeli na tri različite platforme:

- Java standardno izdanje (eng. *Java Standard Edition - Java SE*) - namijenjeno kućnim računalima.

- Java mikro izdanje (eng. *Java Micro Edition - Java ME*) - namijenjeno mobilnim uređajima - dlanovnik, mobitel, itd.

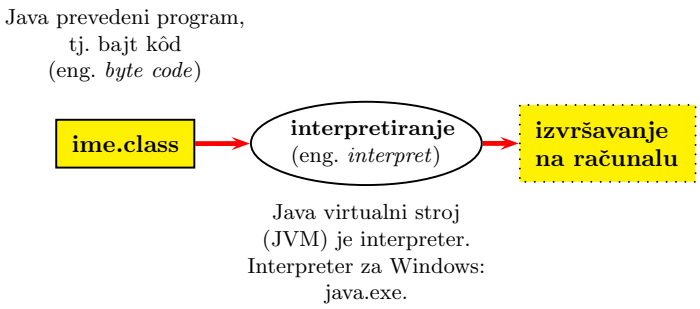
- Java poslovno izdanje (eng. *Java Enterprise Edition - Java EE*) - namijenjena poslužiteljima (eng. *server*), za mrežne, poslovne aplikacije.

2.2.3 Razvoj i izvršavanje Java programa

U ovom slučaju, za izradu simulacije u Javi, potrebno je standardno izdanje Java platforme (Java SE). Ovo izdanje sadrži Java izvedbenu okolinu (JRE) i Java razvojnu opremu (JDK) koje se mogu besplatno preuzeti[1] i koristiti na računalu.

Java izvedbena okolina (JRE)

Java izvedbena okolina (JRE) je potrebna da bi se Java prevedeni programi[2] *mogli izvršavati na nekom računalu.* Za izvršavanje tog programa je potrebno provesti interpretaciju tzv. "bajt kôda" koju obavlja interpreter, tj. Java virtualni stroj (JVM) - slika 2.1. Java izvedbenu okolinu čine Java virtualni stroj (JVM), biblioteka programskih elemenata (Java API) i ostale komponente.



Slika 2.1: Shema izvršavanja Java prevedenog programa.

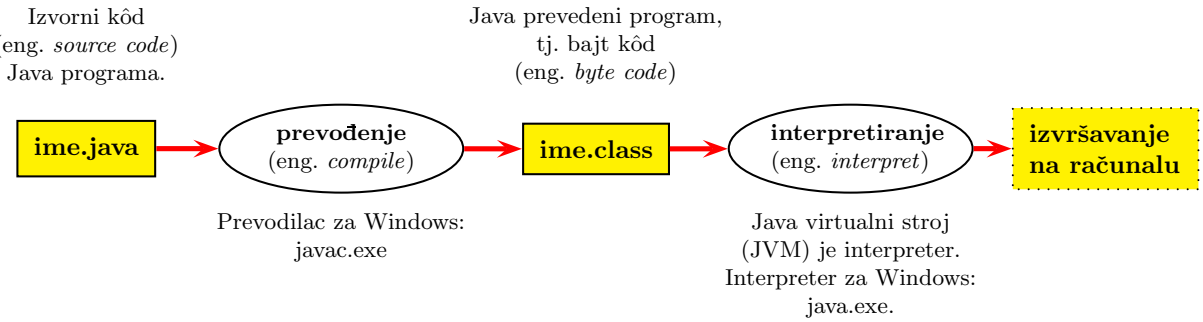
Java razvojna oprema (JDK)

Java razvojna oprema (JDK) služi za razvoj i izvršavanje Java programa, a uključuje Java izvedbenu okolinu (JRE), te razne alate važne za razvoj programa. Pritom je važno napomenuti da nije potrebno zasebno instalirati Java izvedbenu okolinu (JRE). Pri razvoju vrijedi slijedeće: izvorni kôd Java programa[3] se prvo prevodi (eng. *compile*) u bajt kôd, a potom slijedi interpretacija bajt kôda, čime se izvršava Java prevedeni program - slika 2.2.

[1] Hiperveza na stranici 38. *Preuzeti* podrazumijeva eng. *download*.

[2] Java prevedeni (eng. *compile*) program je tzv. "bajt kôd" (eng. *byte code*), koji ima ekstenziju ".class".

[3] Izvorni kôd (eng. *source code*) Java programa ima ekstenziju ".java".



Slika 2.2: Shema razvoja i izvršavanja Java programa.

2.2.4 Neovisnost o hardveru i operacijskom sustavu

U programskim jezicima C, C++, i dr. izvorni kôd se prevodi u tzv. objektni kôd[4] koji je specifičan, ovisan o pojedinom hardveru i operacijskom sustavu (Microsoft Windows, Unix, itd.). Na primjer, nije moguće pokrenuti C++ prevedeni program u Windows-ima, ako je taj program preveden u Unix-u.

Java prevedeni program, odnosno bajt kôd, s druge strane omogućuje neovisnost o operacijskom sustavu i hardveru, uz nužnu instaliranu Java platformu (Java platforma je ovisna o operacijskom sustavu). Ovime je omogućeno da se npr. Java prevedeni program može pokrenuti u Windows-ima, iako je program preveden u Unix-u.

U razvoju programa, kod Jave je potrebno prevođenje, a potom interpretiranje, dok je kod C++-a potrebno samo prevođenje. Evidentno je pritom da je brzina razvoja i izvršavanja veća kod C++ programskog jezika.

2.3 Programiranje u Javi

2.3.1 Prvi Java program

Upoznavanje sa programskim jezicima obično započinje sa jednostavnim programima koji ispisuju poruku "Pozdrav svijetu!" (eng. *Hello world*). Izvorni kôd takvog Java programa glasi (datoteka `PozdravSvijetu.java`):

```
public class PozdravSvijetu {
  public static void main(String[] args) {
    System.out.println("Pozdrav svijetu!");
  }
}
```

[4] Tj. kôd u strojnom jeziku (eng. *machine language code*) ili matični kôd (eng. *native code*). Za Windows operativni sustav, takav prevedeni program obično ima ekstenziju ".exe", a zove se i izvršna datoteka.

Kao što je već rečeno, kako bi se ovaj program izvršio na računalu, prvo je potrebno obaviti prevođenje u konzoli[5]:

```
javac.exe PozdravSvijetu.java
```

Ukoliko je prevođenje prošlo uspješno, kreira se datoteka `PozdravSvijetu.class`. Slijedi interpretiranje:

```
java.exe PozdravSvijetu
```

Program se izvršava, pa se u konzoli ispiše poruka:

```
Pozdrav svijetu!
```

Pojašnjenje izvornog kôda

Ovaj program čini klasa `PozdravSvijetu` koja sadrži metodu `main` (izvršavanje programa započinje pozivom metode `main`). Java program koji sadrži metodu `main` (ta metoda mora biti napisana u tom obliku) se ujedno naziva **Java aplikacija**. Linija kôda `System.out.println("Pozdrav svijetu!");` predstavlja primjenu jedne od mnogih, gotovih softverskih elemenata u biblioteci Java API, ova konkretno ispisuje zadani tekst u konzoli.

Ključna riječ `public` označuje kontrolu pristupa. Takve kontrole (javno - `public`, zaštićeno - `protected`, privatno - `private` i dr.) su važne jer pridonose boljoj konzistentnosti programa. Ovo je ujedno i jedan od principa objektno orijentiranog programiranja koje se naziva **enkapsulacija** (eng. *encapsulation*).

Budući da u ovom programu nije bilo riječi o objektu (tj. objekt nije instanciran), metoda `main` sadrži i ključnu riječ `static`. Treća ključna riječ `void` metode `main` ukazuje da ta metoda ne vraća nikakav rezultat. Dodatak `main` metodi u zagradama, `String[] args` služi za eventualni unos argumenata pri pokretanju programa[6].

2.3.2 Java aplikacija sa grafičkim sučeljem

Osim aplikacije koja se izvršava u konzoli, u Javi se može kreirati i aplikacija sa grafičkim - korisničkim sučeljem (eng. *graphical user interface* - GUI).

Jednostavan primjer izvornog kôda Java aplikacije sa grafičkim sučeljem glasi (datoteka `PozdravSvijetuW.java`):

```
import java.awt.*;
```

[5]Konzola tj. naredbeni prozor (eng. *command prompt*). U Windows operativnom sustavu, konzola se pokreće sa `cmd.exe`.

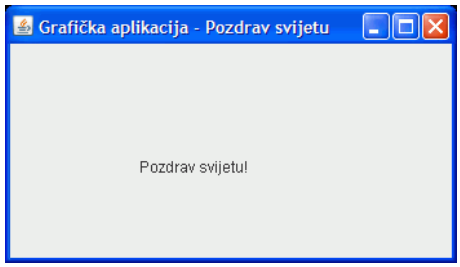
[6]Npr. ako Java aplikacija `zbroj.class` zbraja dva argumenta, tada bi u konzoli napisali sljedeće: `java.exe zbroj 2 3` - time bi se ispisao broj `5`.

```
import javax.swing.*;

public class PozdravSvijetuW {
  public static void main(String[] args) {
    JFrame prozor = new JFrame();
    crtaciPanel panel = new crtaciPanel();
    prozor.add(panel);
    prozor.setSize(350,200);
    prozor.setTitle("Grafička aplikacija - Pozdrav svijetu");
    prozor.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    prozor.setVisible(true);
  }
}

class crtaciPanel extends JPanel {
  public void paint(Graphics g) {
    g.drawString("Pozdrav svijetu!",100,100);
  }
}
```

Nakon uspješna prevođenja i interpretiranja, pojaviti će se slijedeći prozor (slika 2.3).



Slika 2.3: Java aplikacija sa grafičkim sučeljem

Pojašnjenje izvornog kôda

Važno je napomenuti da ova i prethodna aplikacija (kao i sve ostale aplikacije) moraju imati isti naziv datoteke kao i pripadna klasa koja sadrži `main` metodu.

Java API biblioteka sastoji se od mnoštva paketa[7] (koji služe za različite potrebe), dok se paketi sastoje od klasa. Ova aplikacija započinje sa `import` naredbama pomoću kojih se koriste klase paketa `java.awt` i `javax.swing` koji služe za izradu grafičkog sučelja i grafike.

Ovaj program čine dvije klase `PozdravSvijetuW` i `crtaciPanel`. Druga klasa koristi ključnu riječ `extends` koja omogućuje nasljeđivanje, tj. proširenje neke postojeće klase (u ovom slučaju, proširuje se klasa `JPanel`). **Nasljeđivanje** se

[7]Popis svih paketa se može vidjeti u Java API dokumentaciji. Hiperveza u literaturi pod brojem [11].

upotrebljava kako bi se iskoristio već postojeći kôd, čime se povećava brzina izrade programa. Nasljeđivanje (eng. *inheritance*) je također jedan od principa objektno orijentiranog programiranja.

Za razliku od prethodne, u ovoj aplikaciji se instanciraju objekti na koje ukazuju objektne varijable `prozor` i `panel`. Instanciranje se vrši operatorom `new` i pripadnim konstruktorom (tj. metode `JFrame()`, `crtaciPanel()` itd...) koji postavlja svojstva objekta:

```
JFrame prozor = new JFrame();
crtaciPanel panel = new crtaciPanel();
```

Kao što je rečeno, objekti su instance neke klase, koji pritom mogu koristiti metode i podatke te klase. U ovoj aplikaciji metode mijenjaju stanje objekta na koji ukazuje objektna varijabla `prozor`:

```
prozor.setSize(350,200);
prozor.setTitle("Grafička aplikacija - Pozdrav svijetu");
prozor.add(panel);
prozor.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
prozor.setVisible(true);
```

2.3.3 Java applet

Osim Java aplikacije, postoje i Java appleti. Osnovna razlika je da se applet izvršava u okviru internet stranice uz pomoć internet preglednika (npr. Internet Explorer, Mozilla Firefox, itd.). Applet se može pokrenuti i preko naredbe `appletviewer.exe` u konzoli. Na primjer, ukoliko internet stranica `stranica.html`[8] ukazuje na applet `nekiapplet.class`, piše se:

```
appletviewer.exe stranica.html
```

S druge strane, postupak prevođenja appleta je isti kao i za aplikaciju (uporabom naredbe `javac.exe`, spomenut u odjeljku 2.3.1).

Pri izradi appleta, koristi se gotova klasa `JApplet` koja koristi četiri temeljne metode važne za rad appleta:

- Metoda `init` (inicijalizacija). Mjesto inicijaliziranja varijabli, ova metoda se poziva prva.
- Metoda `start` (pokretanje). Ova metoda se poziva nakon `init` metode, a služi za početak izvršavanja appleta.
- Metoda `stop` (zaustavljanje). Metoda služi za zaustavljanje izvršavanja appleta. Poziva se prije `destroy` metode.

[8] Ta internet stranica pritom mora sadržavati kôd:
`<applet code="nekiapplet.class" width=100 height=100> </applet>`.

- Metoda `destroy` (uništavanje). Ova metoda se poziva nakon `stop` metode, kada se gasi internet preglednik ili appletviewer.

Važno je napomenuti da pritom nema `main` metode, što predstavlja još jednu razliku u odnosu na aplikaciju.

2.4 Java elementi korišteni u appletu

U prethodnom odjeljku spomenuti su različiti elementi prisutni u Javi, te principi objektno orijentiranog programiranja (nasljeđivanje i enkapsulacija). No predstavljeni elementi su samo djelić od mnoštva elemenata koje Java pruža. U ovom odjeljku će se spomenuti elementi koji će biti korišteni u appletu.

2.4.1 Komponente i kontejneri

Program sa grafičkim sučeljem u Javi se sastoji od tzv. **komponenata**. Među komponentama se razlikuju tzv. **kontejneri** koji mogu sadržavati druge komponente.

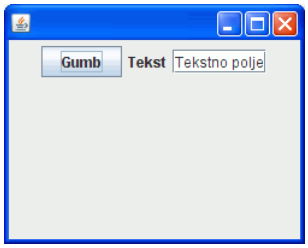
Ukoliko se za grafičko sučelje koristi napredni paket `javax.swing`, tada program mora imati najmanje jedan tzv. kontejner najvišeg nivoa (eng. *top-level container*). Takvi kontejneri su klase `JFrame` (za Java aplikacije), `JDialog` (za dijaloške okvire) i `JApplet` (za Java applete). Na kontejnere najvišeg nivoa se obično dodavaju drugi kontejneri poput `JPanel`, `JInternalFrame` radi efikasnijeg rasporeda komponenata.

Postoji mnoštvo komponenata koje se mogu dodati npr. `JButton` (gumb), `JLabel` (natpis), `JTextField` (tekstno polje) i mnogi drugi. Dodavanje komponente na kontejner se vrši metodom `add`.

Jednostavan primjer dodavanja komponenata:

```
JFrame prozor = new JFrame();
JPanel panel = new JPanel();
panel.add(new JButton("Gumb"));
panel.add(new JLabel("Tekst"));
panel.add(new JTextField("Tekstno polje"));
prozor.add(panel);
```

Dakle, prvo se kreira kontejner najvišeg nivoa (`prozor`), a potom kontejner `panel`. Metodom `add` se dodavaju različite komponente u kontejner `panel`, a potom se kontejner `panel` dodaje na kontejner najvišeg nivoa `prozor`. Rezultat bi predstavljao sljedeći prikaz (slika 2.4).



Slika 2.4: Komponente i kontejneri

2.4.2 Crtanje - uporaba grafike

Za crtanje grafičkih oblika u Javi, koristi se metoda `paint (Graphics g)`. U ovoj metodi objektna varijabla `g` tipa `Graphics` ukazuje na objekt na koji će se dodavati grafički oblici.

Sa razvojem Jave, pojavila su se poboljšanja u korištenju grafike (pojava nove klase `Graphics2D`). Kako bi se mogli koristiti te napredne mogućnosti, potrebno je uvesti objektnu varijablu tipa `Graphics2D` koja će se povezati sa spomenutom varijablom `g`:

```
Graphics2D g2 = (Graphics2D) g;
```

Na objekt na koji ukazuje varijabla `g2` moguće je dodati različite grafičke oblike (ravna linija, pravokutnik, elipsa, kružni luk itd...)[9]. Dodavanje grafičkih oblika se vrši preko metode `draw` (crta samo linije grafičkih oblika) ili preko metode `fill` (iscrtava unutrašnjost oblika).

Primjer korištenja grafike (samo metoda `paint`):

```
public void paint(Graphics g)
{
  Graphics2D g2 = (Graphics2D) g;

  g2.draw(new Line2D.Double(50,50,80,100));
  g2.fill(new Rectangle2D.Double(150,50,50,50));
  g2.draw(new Ellipse2D.Double(250,50,150,50));
}
```

Parametri korišteni za ove grafičke oblike određuju njihov smještaj i veličinu. Rezultat ove metode bi predstavljao sljedeći prikaz (slika 2.5).

2.4.3 Omogućavanje animacije

U Javi postoji više načina koji omogućuju animaciju odnosno brzo izmjenjivanje slika. Za animaciju appleta korištena je dretva, koja sadrži animacijsku petlju u kojoj se održava pravilna izmjena slika odnosno osvježavanje zaslona.

[9]Popis grafičkih oblika se može vidjeti u Java API dokumentaciji, u paketu `java.awt.geom`.

Slika 2.5: Uporaba grafičkih oblika

Kako bi dretvu i animacijsku petlju implementirali u applet, potrebno je napraviti sljedeće:

- Dodati ključne riječi `implements Runnable` u proširenu JApplet klasu (kojom se omogućuje animacijska petlja):

```
class CoriolisApplet extends JApplet implements Runnable
```

- U `start` metodi appleta kreirati, a potom omogućiti pokretanje dretve (prije toga je potrebno deklarirati varijablu `dretva` kao podatak proširene JApplet klase - `Thread dretva;`):

```
dretva = new Thread(this);
dretva.start();
```

- Implementirati animacijsku petlju, tj. `run` metodu u appletu:

```
public void run() {
  long tm = System.currentTimeMillis();
  while(Thread.currentThread()== dretva) {
    drugipanel.repaint();
    tm = tm + delay ;
    try {
      Thread.sleep(Math.max(0, tm - System.currentTimeMillis()));
    }
    catch(InterruptedException e) {
      System.err.println("Greska u run() metodi");
    }
  }
}
```

Dakle pokretanjem dretve, pokreće se i animacijska petlja. U ovoj petlji se prvo osvježava zaslon (metodom `repaint`), a potom slijedi čekanje, tj. vremenski interval između slika (metodom `sleep`, kojom se dretva stavlja u stanje spavanja). Ova petlja se izvršava sve dok se applet izvršava.

2.4.4 Omogućavanje interaktivnosti

Interaktivni programi sa grafičkim sučeljima su određeni događajima, npr. pritisak tipke na tipkovnici, pritisak gumba u programu, gibanje miša i dr. koji omogućuju korisniku upravljanje programom.

Kako bi se omogućila interaktivnost u Javi, prvo je potrebno izvor događaja (npr. gumb) registrirati sa slušačem događaja (eng. *event listener*) preko kojeg će se pozvati metoda slušača, koja radi određenu zadaću.

Na primjer, kreirati će se gumb i omogućiti njegova interaktivnost:

- Zbog tražene interaktivnosti, potrebno je koristiti paket `java.awt.event`:

```
import java.awt.event.*;
```

- Kreiranje izvora događaja (gumb):

```
gumb = new JButton("Pritisni gumb...");
```

- Registracija izvora događaja sa slušačem:

```
gumb.addActionListener(this);
```

- U klasi u kojoj se nalazi izvor događaja, treba navesti ključne riječi `implements ActionListener`, koja omogućuje metodu slušača `actionPerformed`:

```
public void actionPerformed(ActionEvent e) {
  if(e.getSource() == gumb) {
    gumb.setText("Promjena!");
  }
}
```

Ukoliko se pritisne gumb, izvršava se metoda slušača (u ovom slučaju, promijeniti će se tekst na gumbu). Od ovog primjera, analogno se može dobiti interaktivnost različitih izvora događaja.

2.5 Upute za korištenje appleta

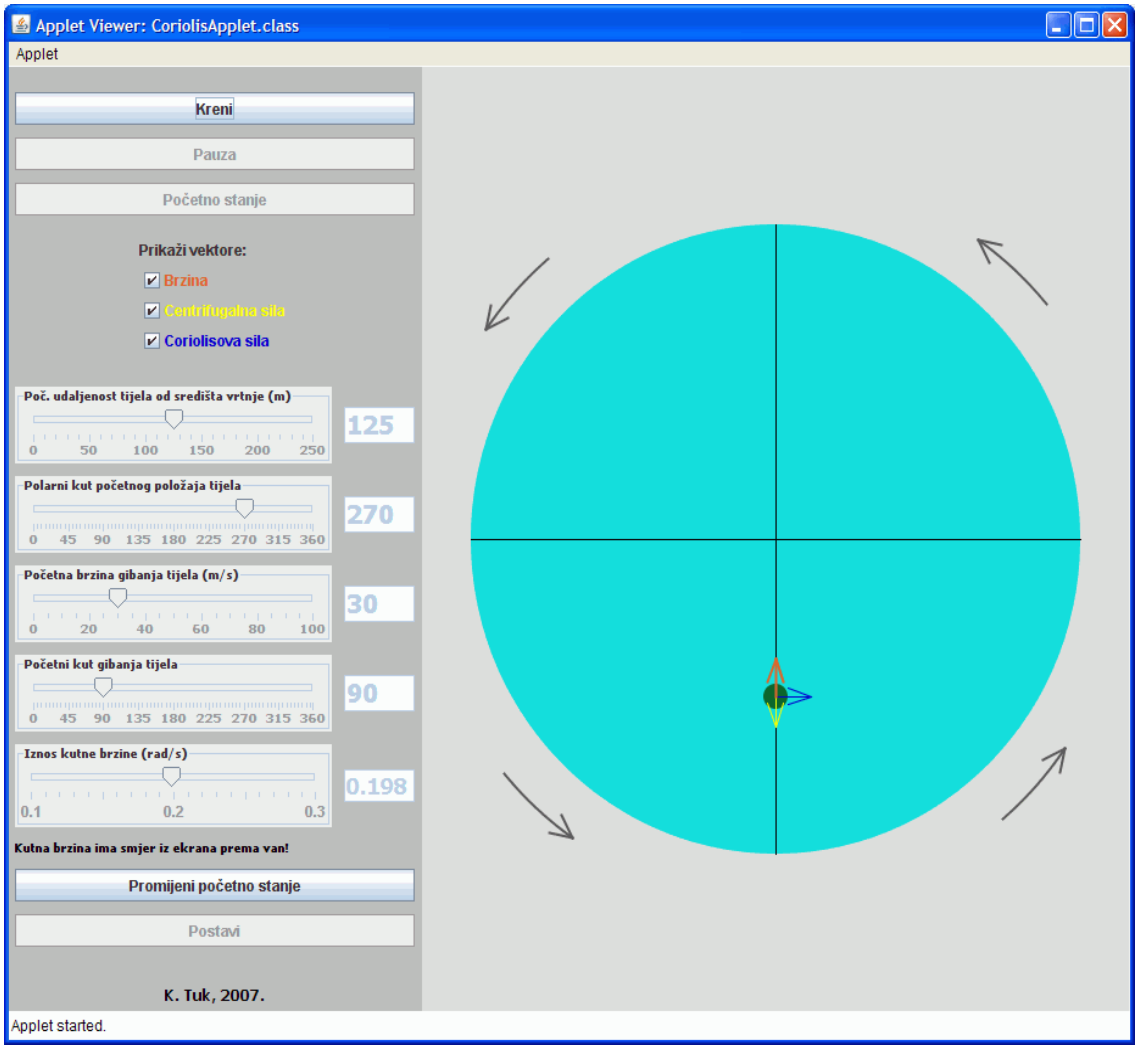
U ovom odjeljku, napomenut ću upute i značenje appleta. Kada se applet pokrene, na lijevoj strani appleta se nalazi kontrolna ploča, a na desnoj strani okvir u kojem se događa simulacija - tijelo koje se giba u sustavu koji se vrti (npr. vrtuljak, svemirski kotač, itd.).

Na kontrolnoj ploči se na vrhu nalaze tri gumba: **Kreni** (započinje ili **Nastavi** koji nastavlja simulaciju), **Pauza** (zaustavlja simulaciju) i **Početno stanje** (simulacija se postavlja u početno stanje koje određuju zadani uvjeti).

Ispod ovih gumba se nalaze okviri za izbor kojima se omogućuje ili onemogućuje prikaz danih vektora u samoj simulaciji. Na početku su svi vektori prikazani.

Niže od okvira za izbor se nalaze pet pomičnih skala koje omogućuju promjenu početnih uvjeta. No kako bi mogli mijenjati početne uvjete, prije toga se mora pritisnuti gumb **Promijeni početno stanje**, a potom potvrditi promjene sa gumbom **Postavi** (ova dva gumba se nalaze ispod pet pomičnih skala).

Prva pomična skala može mijenjati početnu udaljenost od središta vrtnje (u metrima). Ispod slijedi pomična skala koja određuje polarni kut, koji uz početnu udaljenost od središta vrtnje određuje početni položaj tijela. Sljedeća pomična skala određuje iznos početne brzine tijela koje se giba u sustavu koji se vrti. Slijedi pomična skala koja predstavlja polarni kut koji određuje smjer početne brzine tijela



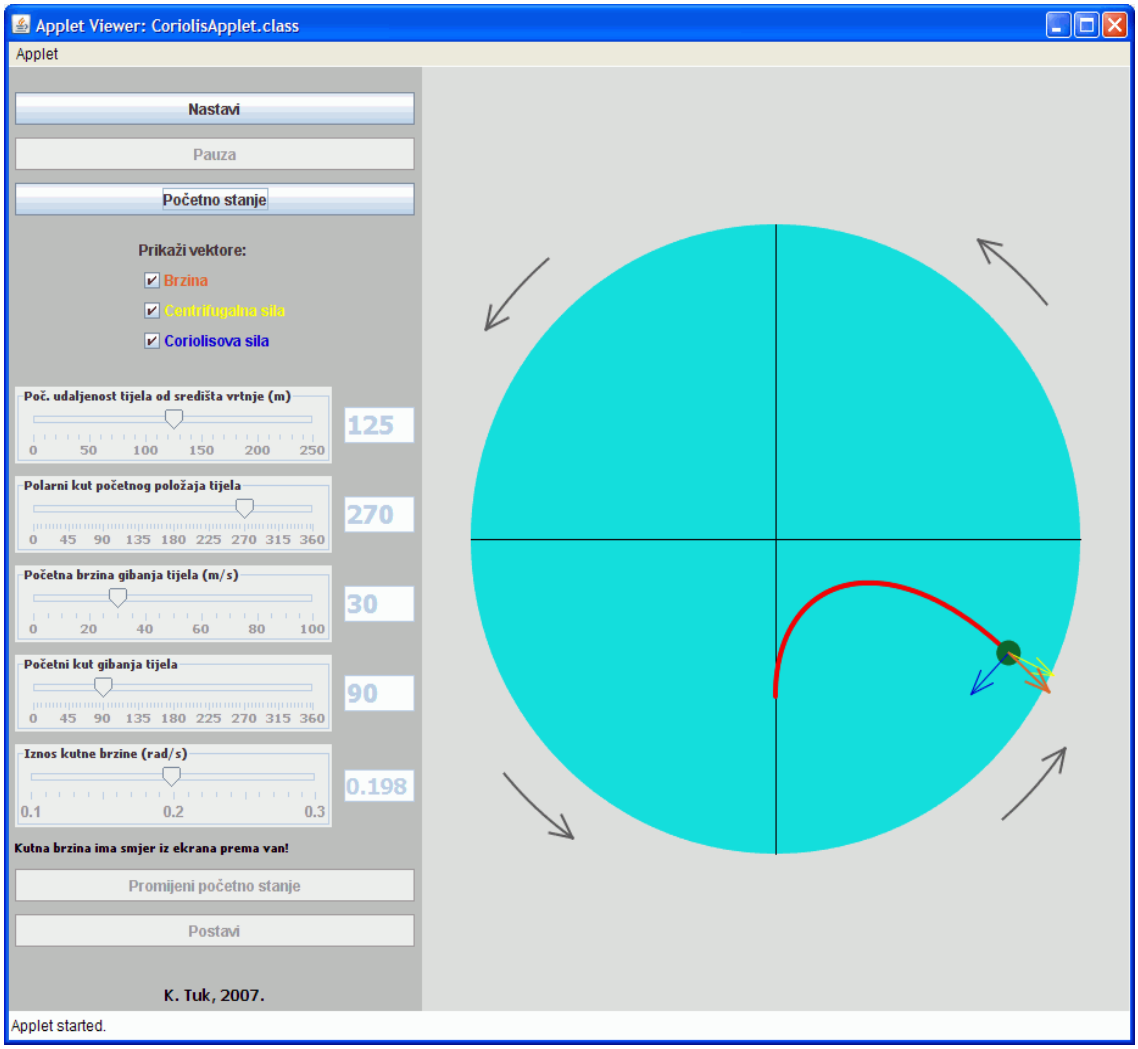
Slika 2.6: Applet u početnom trenutku

koje se giba u sustavu koji se vrti. Zadnja pomična skala određuje iznos kutne brzine kojom sustav rotira. Ispod ove skale je naznačena napomena, da vektor kutne brzine ima smjer iz ekrana prema van - ili oznaka $\odot$.

Pri pomičnim skalama desno se ujedno nalaze i tekstni okviri koji prikazuju odabrane vrijednosti, a moguće je i upisivanje vrijednosti.

Zadane, početne vrijednosti su:
$R = 250\,m$ - Polumjer sustava koji rotira.
$r = 125\,m$ - Početnu udaljenost tijela od središta vrtnje (radijus-vektor).
$\theta = 270°$ - Polarni kut pomoću kojeg se određuje početni položaj tijela.
$v = 30\,m/s$ - Iznos početne brzine tijela koje se giba u sustavu koji se vrti.
$\alpha = 90°$ - Polarni kut koji određuje smjer početne brzine v.
$\omega = 0,198\,rad/s$ - Kutna brzina rotiranja sustava $\omega = \sqrt{\frac{g}{R}} = 0.198\,rad/s$ ($g = 9.81 m/s^2$ je akceleracija težne sile).

Prilikom pokretanja simulacije, ukoliko se tijelo giba, ostavlja se crveni trag koji predstavlja putanju tijela u sustavu koji se vrti.



Slika 2.7: Zaustavljeni applet

Implementacija Java appleta u nastavi fizike

Priprava za nastavnu jedinicu

Škola: Gimnazija

Razred: 1.

Nastavni predmet: Fizika

Nastavna cjelina: Sila i gibanje

Nastavna jedinica: Inercijski i neinercijski sustav, Referentni sustav koji se vrti

Trajanje: 1 školski sat

Mjesto izvođenja nastave: učionica

Cilj nastavne jedinice: Upoznati učenike sa inercijskim i neinercijskim sustavima, te značenje centrifugalne i Coriolisove sile.

Zadaci nastave:

- Materijalni (obrazovni) - Definirati pojam inercijskog i neinercijskog sustava, poznavati razliku između centripetalne i centrifugalne sile, poznavati posljedice Coriolisove sile.
- Funkcionalni - Razvijanje intelektualnih i misaonih sposobnosti učenika, otkrivanje i spoznavanje veza i odnosa.
- Odgojni - Stvaranje pozitivne atmosfere u razredu, razvijanje prihvaćanja argumenata drugih učenika.

Oblici rada: Frontalni, individualni.

Nastavne metode: Metoda razgovora, metoda demonstracije, rad na računalu

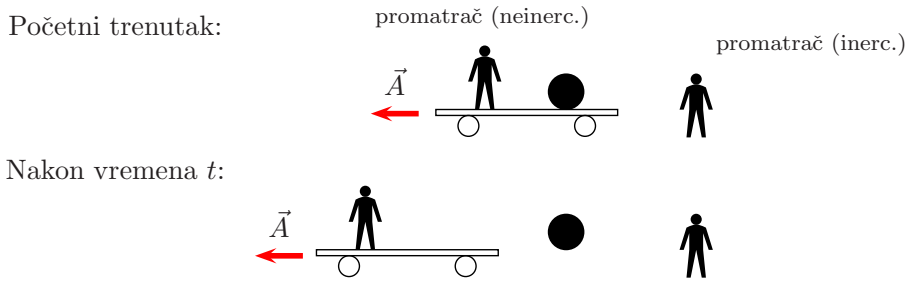
Nastavna sredstva i pomagala: Kolica (autić), kugla, tijelo u obliku kvadra, kružna ploča sa šipkom, tanjurić, lijevak, posuda s vodom, centrifugalni stroj, računalo sa instaliranom Java platformom, Java applet, projektor, kreda, ploča.

Korelacija: Informatika (programiranje), geografija (skretanje morskih i zračnih struja).

Artikulacija sata:
Što govori 1. Newtonov zakon? Dakle, slobodno tijelo miruje ili se giba jednoliko po pravcu. Referentni sustavi u kojima vrijedi 1. Newtonov zakon su inercijski sustavi.

Da li postoje sustavi koji nisu inercijski i kakve su njihove značajke (primjer)? Neinercijski sustav se ne giba jednoliko, već akcelerira u odnosu na inercijski sustav. Primjer dizalo, automobil koji akceleriraju.

Pokus (slika 2.8) - kuglica mase m na kolicima (kolica akceleriraju ulijevo sa akceleracijom $\vec{A}$), jedan promatrač na kolicima, drugi promatrač u inercijskom sustavu. Kako promatrači doživljavaju gibanje kuglice? Promatrač u inercijskom sustavu - kuglica miruje. Promatrač u neinercijskom sustavu utvrđuje da kuglica akcelerira udesno (suprotno od smjera akceleracije kolica). Prema tome u tom sustavu djeluje tzv. inercijska sila.
$\vec{F}_{in} = -m \cdot \vec{A}$



Slika 2.8: Pokus - Dva promatrača i kuglica.

U neinercijskom sustavu koristi se 2. Newtonov zakon $\vec{F} = m \cdot \vec{a}$ za neko tijelo na koje djeluje sila. I u neinercijskom sustavu se može koristiti 2. Newtonov zakon, ali uz dodatak inercijske sile: $\vec{F} + \vec{F}_{in} = m \cdot \vec{a}$, tj. $\vec{F} - m \cdot \vec{A} = m \cdot \vec{a}$.

Razmotriti primjer: osoba u dizalu.

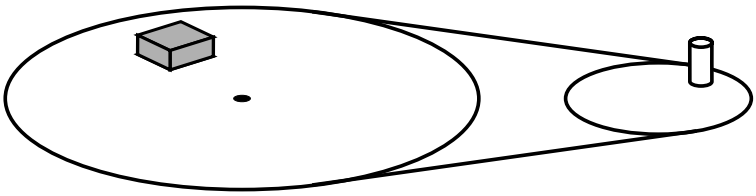
U nastavku slijedi razmatranje sustava koji se vrti. Kada se osoba nalazi na vrtuljku, da li se pritom osjeća kakva sila? U kojem smjeru djeluje ta sila? Iskustvo nam govori da pritom djeluje sila koja je usmjerena od središta vrtnje prema van.

No što će reći opažač koji stoji mirno? Uz podsjećanje na prošlo predavanje, na tijelo djeluje centripetalna sila $F_{cp} = \frac{mv^2}{r}$, koja je usmjerena prema središtu vrtnje.

Pokus (slika 2.9) - tijelo postavimo na kružnu ploču koja rotira (analiza).

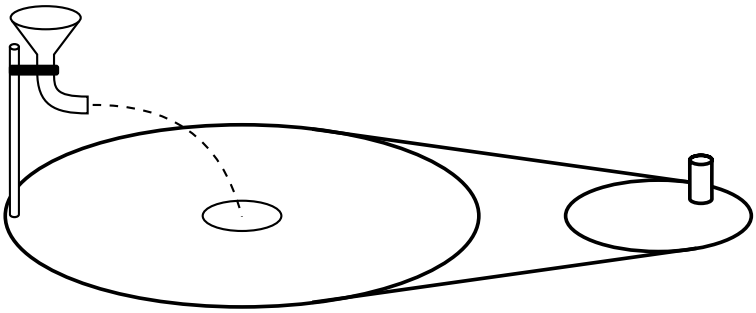
Što smo utvrdili za tijelo koje se promatra u neinercijskom sustavu? U tom sustavu, na tijelo djeluje inercijska sila $\vec{F}_{in} = -m \cdot \vec{A}$, tj. $\vec{F}_{in} = -\frac{mv^2}{r}$, a to je upravo ona sila koja djeluje na osobu na vrtuljku. Tu silu nazivamo centrifugalna sila. Posebna napomena - gdje djeluje koja sila.

Prelazimo na situaciju kada se tijelo giba u sustavu koji rotira. Pokus (slika 2.10) - centrifugalni stroj, lijevak, tanjurić, posuda s vodom, kružna ploča sa



Slika 2.9: Pokus - Tijelo na rotirajućoj kružnoj ploči (koristeći centrifugalni stroj).

šipkom. Pritom se može koristiti i Java applet koji je napravljen u tu svrhu. Promatra se putanja (smjer mlaza vode), u različitim uvjetima (rasprava).



Slika 2.10: Pokus - Promatra se smjer mlaza vode koju puštamo kroz lijevak (kružna ploča rotira) u ovom postavu pokusa.

Dakle, mlaz skreće (promatrajući u sustavu koji se vrti) u desno ako se ploča vrti u smjeru suprotnom kazaljke na satu, odnosno u lijevo ako se ploča vrti u smjeru kazaljke na satu. Promatrač u sustavu koji vrti uočava dakle dodatnu inercijsku silu na tijelo koje se giba u toms sustavu, a koja se naziva Coriolisova sila.

Primjer Coriolisove sile - anticikloni, morske struje na Zemlji.

Ponavljanje današnjeg predavanja (5 minuta).

Zaključak

U sadašnje vrijeme, kada se računalna tehnologija stalno razvija, pružaju se brojne mogućnosti koje mogu poboljšati nastavu fizike u školama i fakultetima, a ujedno i više popularizirati fiziku. Na primjer, dostupnost i raširenost digitalnih knjiga (eng. *e-book*), materijala, slika, snimljeni video-zapis fizikalne pojave ili eksperimenta, sustavi za mjerenje, detektori, interaktivne animacije (simulacije) itd... U konkretnom slučaju, ukoliko fizikalni eksperiment nije moguće izvesti ili ako je rezultat eksperimenta učenicima odnosno studentima nejasan, interaktivna animacija može poslužiti kao dobra zamjena.

Kroz ovu simulaciju fizikalnog problema, želim potaknuti i druge, ponajprije kolege studente na uporabu neke od računalnih tehnologija, jer osim nadogradnje znanja iz fizike i informatike, nude se i šire mogućnosti zapošljenja.

Literatura

[1] R. Krsnik: *Fizika 1, Udžbenik za nastavu fizike u 1. razredu gimnazije,* II. izdanje, Školska knjiga, Zagreb, 1996.

[2] D. Morin: *Introductory Classical Mechanics, with Problems and Solutions,* 2004. http://www.physics.harvard.edu/people/facpages/morin.html

[3] A. Persson: *History of Meteorology 2 - The Coriolis Effect: Four centuries of conflict between common sense and mathematics,* 2005. http://www.meteohistory.org/2005historyofmeteorology2/01persson.pdf

[4] H. Goldstein, C. Poole, J. Safko: *Classical mechanics,* 3. izdanje, Addison-Wesley, 2000.

[5] C. S. Horstmann, G. Cornell: *Core Java 2 Volume I - Fundamentals,* 7. izdanje, Prentice Hall, 2004.

[6] S. Zakhour, S. Hommel, J. Royal, I. Rabinovitch, T. Risser, M. Hoeber: *The Java Tutorial Fourth Edition: A Short Course on the Basics,* 4.izdanje, Addison-Wesley, 2006.

[7] D. Flanagan: *Java in a Nutshell,* 5. izdanje, O'Reilly, 2005.

[8] R. Morelli, R. Walde: *Java, Java, Java: Object-Oriented Problem Solving,* 3. izdanje, Prentice Hall, 2005.

[9] H. Schildt: *Java 2: The Complete Reference,* 5. izdanje, McGraw-Hill, 2004.

[10] G. Muić, M. Jurak: *Računarski praktikum 2,* PMF-matematički odjel, 2006. http://web.math.hr/nastava/rp2

[11] *JDK 6 Documentation (included Java API documentation),* 2006. http://java.sun.com/javase/6/docs

[12] *Java Tutorials,* 2007. http://java.sun.com/docs/books/tutorial/index.html

[13] M. Kosović: *Izrada interaktivne animacije u Java-i za simuliranje gibanja nabijene čestice u homogenom magnetskom polju,* Zagreb, 2005.

[14] *Wikipedia, the free encyclopedia,* http://en.wikipedia.org

Korištene aplikacije

Za izradu ovog diplomskog rada korištene su aplikacije:

- MiKTeX - TeX/LaTeX distribucija, pomoću kojeg je ovaj rad napisan
  http://www.miktex.org
- PSTricks - alat za vektorsko crtanje, uključen u MiKTeX distribuciju
  http://tug.org/PSTricks/main.cgi
- Inkscape - alat za vektorsko crtanje
  http://www.inkscape.org
- Ghostscript, GSView - postscript alati
  http://www.cs.wisc.edu/ ghost
- Java platforma, standardno izdanje 6 (Java SE 6) -
  Java izvedbena okolina (JRE), Java razvojna oprema (JDK)
  http://java.sun.com/javase/downloads/index.jsp

Korišten je Microsoft Windows XP Professional, Version 2002, Service Pack 2 kao operativni sustav.

Prilozi

Prilog - kôd Java appleta

```
//Krešimir Tuk, 2007.
//
//Java applet - simulacija gibanja tijela u sustavu koji se vrti (vrtuljak,
//svemirski kotač...)
//za uspješno prevođenje te interpretaciju ovog applet-a, potrebno je instalirati
//Java JDK 6.0 (ili novije verzije)
//

import java.awt.*;
import java.awt.geom.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.ChangeEvent;
import javax.swing.event.ChangeListener;
import javax.swing.border.*;
import java.util.Hashtable;
import java.util.ArrayList;

public class CoriolisApplet extends JApplet implements Runnable, ActionListener{

  Graphics2D g2;
  JPanel prvipanel, prikazivektore;
  crtacipanel drugipanel;
  JButton pocni, pauza, pocetno_stanje, promijeni_poc_stanje, postavi_stanje;
  JLabel autor, napomena;
  Thread dretva;
  ArrayList<Coord> putanja = new ArrayList<Coord>();
  JSlider Slider_r0, Slider_fi0, Slider_V0, Slider_alfa0, Slider_omega;
  JTextField Tekstpolje_r0, Tekstpolje_fi0, Tekstpolje_V0, Tekstpolje_alfa0,
Tekstpolje_omega;
  JCheckBox brzina, centrif, coriol;

  int brojac, delay, fps;
  int OMEGA_MIN, OMEGA_MAX, OMEGA_INIT;
  boolean checkPaint = false;

  double r = 250, rTIJ, rPUT, rSMJ, omega, omega0 = Math.sqrt(9.81/r), deltat, temp;
  double X0, Y0, Vx0, Vy0, Ax0, Ay0, V0, alfa0, fi0, r0;
  double X1, Y1, Vx1, Vy1, Ax1, Ay1, V1, alfa1, fi1, r1;
  double Xcor, Ycor, Xcf, Ycf;
  double POCRADIJUS = 0.5 * r, POCFI = Math.toRadians(270), POCBRZINA = 30,
```

```
POCALFA = Math.toRadians(90);


  //metoda INIT koja sluzi da se postave vrijednosti velicina
  public void init(){

    fps = 30;
    delay = 1000/fps;
    brojac = 0;
    checkPaint = false;

    r0 = POCRADIJUS;
    rTIJ = 10;
    rPUT = 2;
    rSMJ = r + 40;
    omega = omega0;
    deltat = 0.05;
    V0=30;
    OMEGA_MIN = 100;
    OMEGA_MAX = 300;
    OMEGA_INIT = (int)(omega0*1000);

    fi0 = POCFI; alfa0 = POCALFA;
    X0 = r0 * Math.cos(fi0); Y0 = r0 * Math.sin(fi0);
    Vx0 = V0 * Math.cos(alfa0); Vy0 = V0 * Math.sin(alfa0);
    temp = Math.atan2(Vy0,Vx0) + Math.toRadians(270);
    Xcor = V0 * omega * Math.cos(temp) * 5;
    Ycor = V0 * omega * Math.sin(temp) * 5;
    Xcf = (omega*omega)*r0*Math.cos(fi0) * 5;
    Ycf = (omega*omega)*r0*Math.sin(fi0) * 5;
    Ax0=V0 * omega * Math.cos(temp) + (omega*omega)*r0*Math.cos(fi0);
    Ay0=V0 * omega * Math.sin(temp) + (omega*omega)*r0*Math.sin(fi0);

    putanja.add(new Coord(X0, Y0));

  }

  public void start(){

    //postavljanje velicine appleta
    Dimension velicina = new Dimension(920,750);
    setMinimumSize(velicina);

    //postavljanje izgleda appleta u koje ce se smjestiti paneli
    setLayout(new GridBagLayout());
    GridBagConstraints izgledappleta = new GridBagConstraints();

    //postavke polozaja za prvi objekt na appletu - prvipanel
    izgledappleta.gridx=0;
    izgledappleta.gridy=0;
    izgledappleta.weightx =0.5;
    izgledappleta.weighty=1.0;
    izgledappleta.fill=GridBagConstraints.BOTH;

    //PRVIPANEL
    prvipanel = new JPanel();
```

```
prvipanel.setBackground(new Color(188,188,188));

//postavke polozaja u prvom panelu
prvipanel.setLayout(new GridBagLayout());
GridBagConstraints izgledpanela = new GridBagConstraints();

//PRVI OBJEKT - GUMB "KRENI"
izgledpanela.gridx = 0;
izgledpanela.gridy = 0;
izgledpanela.weightx=1;
izgledpanela.gridwidth=2;
izgledpanela.fill = GridBagConstraints.HORIZONTAL;
izgledpanela.insets = new Insets(20,5,5,5);
pocni = new JButton("Kreni");
pocni.addActionListener(this);
prvipanel.add(pocni,izgledpanela);

//DRUGI OBJEKT - GUMB "PAUZA"
izgledpanela.gridx = 0;
izgledpanela.gridy = 1;
izgledpanela.insets = new Insets(5,5,5,5);
pauza = new JButton("Pauza");
pauza.addActionListener(this);
pauza.setEnabled(false);
prvipanel.add(pauza,izgledpanela);

//TRECI OBJEKT - GUMB "POCETNO STANJE"
izgledpanela.gridx = 0;
izgledpanela.gridy = 2;
izgledpanela.insets = new Insets(5,5,5,5);
pocetno_stanje = new JButton("Početno stanje");
pocetno_stanje.addActionListener(this);
pocetno_stanje.setEnabled(false);
prvipanel.add(pocetno_stanje,izgledpanela);

//PRIKAZ VEKTORA NA CRTEZU
izgledpanela.gridx = 0;
izgledpanela.gridy = 3;
izgledpanela.insets = new Insets(10,5,5,5);
izgledpanela.fill = GridBagConstraints.NONE;
prikazivektore = new JPanel();
prikazivektore.setBackground(new Color(188,188,188));
prikazivektore.setLayout(new GridLayout(4, 1));
JLabel naslov_prikazi = new JLabel("Prikaži vektore:");
prikazivektore.add(naslov_prikazi);
brzina = new JCheckBox("Brzina",true);
brzina.setForeground(new Color(225,101,41));
brzina.setBackground(new Color(188,188,188));
centrif = new JCheckBox("Centrifugalna sila",true);
centrif.setForeground(new Color(255,255,0));
centrif.setBackground(new Color(188,188,188));
coriol = new JCheckBox("Coriolisova sila",true);
coriol.setForeground(new Color(0,0,212));
coriol.setBackground(new Color(188,188,188));
prikazivektore.add(brzina);
prikazivektore.add(centrif);
```

```
    prikazivektore.add(coriol);
    prvipanel.add(prikazivektore,izgledpanela);
    izgledpanela.fill = GridBagConstraints.HORIZONTAL;


    //CETVRTI OBJEKT - POMICNA SKALA "POCETNA UDALJENOST OD SREDISTA VRTNJE"
    izgledpanela.gridx = 0;
    izgledpanela.gridy = 4;
    izgledpanela.insets = new Insets(20,5,5,5);
    izgledpanela.gridwidth=1;
    Slider_r0 = new JSlider();
    Slider_r0.setEnabled(false);
    TitledBorder naslovSlider_r0 = new TitledBorder("Poč. udaljenost tijela od
središta vrtnje (m)");
    naslovSlider_r0.setTitleFont(new Font("Tahoma",Font.BOLD,10));
    Slider_r0.setToolTipText("Ova skala određuje radijus-vektor početnog položaja
tijela, u polarnom sustavu.");
    Slider_r0.setBorder(naslovSlider_r0);
    Slider_r0.setFont(new Font("Tahoma",Font.BOLD,9));
    Slider_r0.setMinorTickSpacing(10);
    Slider_r0.setPaintLabels(true);
    Slider_r0.setPaintTicks(true);
    Slider_r0.setMajorTickSpacing(50);
    Slider_r0.setMaximum((int)r);
    Slider_r0.setValue((int)r0);
    Slider_r0.addChangeListener(new ChangeListener() {
      public void stateChanged(ChangeEvent changeevent)
      {
        Promijeni_r0(changeevent);
      }
    });
    prvipanel.add(Slider_r0,izgledpanela);


    //PETI OBJEKT - TEKSTNO POLJE "POCETNA UDALJENOST OD SREDISTA VRTNJE"
    izgledpanela.gridx = 1;
    izgledpanela.gridy = 4;
    izgledpanela.insets = new Insets(20,5,5,5);
    Tekstpolje_r0 = new JTextField();
    Tekstpolje_r0.setEnabled(false);
    Tekstpolje_r0.setColumns(1);
    Tekstpolje_r0.setToolTipText("Pritisni ENTER nakon upisanog broja.");
    Tekstpolje_r0.setText(String.valueOf((int)r0));
    Tekstpolje_r0.setFont(new Font("Tahoma",Font.BOLD,20));
    Tekstpolje_r0.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent actionevent)
      {
        Tekstpolje_r0ActionPerformed(actionevent);
      }
    });
    prvipanel.add(Tekstpolje_r0,izgledpanela);


    //SESTI OBJEKT - POMICNA SKALA "POLARNI KUT POCETNOG POLOZAJA TIJELA"
    izgledpanela.gridx = 0;
    izgledpanela.gridy = 5;
```

```
    izgledpanela.insets = new Insets(5,5,5,5);
    Slider_fi0 = new JSlider();
    Slider_fi0.setEnabled(false);
    TitledBorder naslovSlider_fi0 = new TitledBorder("Polarni kut početnog
položaja tijela");
    naslovSlider_fi0.setTitleFont(new Font("Tahoma",Font.BOLD,10));
    Slider_fi0.setToolTipText("U polarnom sustavu, uz radijus-vektor, polarni
kut određuje položaj tijela. Polarni kut se računa od pozitivne x-osi.");
    Slider_fi0.setBorder(naslovSlider_fi0);
    Slider_fi0.setFont(new Font("Tahoma",Font.BOLD,9));
    Slider_fi0.setMinorTickSpacing(5);
    Slider_fi0.setPaintLabels(true);
    Slider_fi0.setPaintTicks(true);
    Slider_fi0.setMajorTickSpacing(45);
    Slider_fi0.setMaximum(360);
    Slider_fi0.setValue((int)Math.toDegrees(fi0));
    Slider_fi0.addChangeListener(new ChangeListener() {
      public void stateChanged(ChangeEvent changeevent)
      {
        Promijeni_fi0(changeevent);
      }

    });
    prvipanel.add(Slider_fi0,izgledpanela);


    //SEDMI OBJEKT - TEKSTNO POLJE "POLARNI KUT POCETNOG POLOZAJA TIJELA"
    izgledpanela.gridx = 1;
    izgledpanela.gridy = 5;
    izgledpanela.insets = new Insets(5,5,5,5);
    Tekstpolje_fi0 = new JTextField();
    Tekstpolje_fi0.setEnabled(false);
    Tekstpolje_fi0.setColumns(1);
    Tekstpolje_fi0.setToolTipText("Pritisni ENTER nakon upisanog broja.");
    Tekstpolje_fi0.setText(String.valueOf((int)Math.toDegrees(fi0)));
    Tekstpolje_fi0.setFont(new Font("Tahoma",Font.BOLD,20));
    Tekstpolje_fi0.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent actionevent)
      {
        Tekstpolje_fi0ActionPerformed(actionevent);
      }
    });
    prvipanel.add(Tekstpolje_fi0,izgledpanela);


    //OSMI OBJEKT - POMICNA SKALA "POCETNA BRZINA GIBANJA TIJELA"
    izgledpanela.gridx = 0;
    izgledpanela.gridy = 6;
    izgledpanela.insets = new Insets(5,5,5,5);
    Slider_V0 = new JSlider();
    Slider_V0.setEnabled(false);
    TitledBorder naslovSlider_V0 = new TitledBorder("Početna brzina gibanja
tijela (m/s)");
    naslovSlider_V0.setTitleFont(new Font("Tahoma",Font.BOLD,10));
    Slider_V0.setToolTipText("IZNOS početne brzine gibanja tijela.");
    Slider_V0.setBorder(naslovSlider_V0);
```

```
    Slider_V0.setFont(new Font("Tahoma",Font.BOLD,9));
    Slider_V0.setMinorTickSpacing(5);
    Slider_V0.setPaintLabels(true);
    Slider_V0.setPaintTicks(true);
    Slider_V0.setMajorTickSpacing(20);
    Slider_V0.setMaximum(100);
    Slider_V0.setValue((int)V0);
    Slider_V0.addChangeListener(new ChangeListener() {
      public void stateChanged(ChangeEvent changeevent)
      {
        Promijeni_V0(changeevent);
      }
    });
    prvipanel.add(Slider_V0,izgledpanela);


    //DEVETI OBJEKT - TEKSTNO POLJE "POCETNA BRZINA GIBANJA TIJELA"
    izgledpanela.gridx = 1;
    izgledpanela.gridy = 6;
    izgledpanela.insets = new Insets(5,5,5,5);
    Tekstpolje_V0 = new JTextField();
    Tekstpolje_V0.setEnabled(false);
    Tekstpolje_V0.setColumns(1);
    Tekstpolje_V0.setToolTipText("Pritisni ENTER nakon upisanog broja.");
    Tekstpolje_V0.setText(String.valueOf((int)V0));
    Tekstpolje_V0.setFont(new Font("Tahoma",Font.BOLD,20));
    Tekstpolje_V0.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent actionevent)
      {
        Tekstpolje_V0ActionPerformed(actionevent);
      }
    });
    prvipanel.add(Tekstpolje_V0,izgledpanela);


    //DESETI OBJEKT - POMICNA SKALA "POCETNI KUT GIBANJA TIJELA"
    izgledpanela.gridx = 0;
    izgledpanela.gridy = 7;
    izgledpanela.insets = new Insets(5,5,5,5);
    Slider_alfa0 = new JSlider();
    Slider_alfa0.setEnabled(false);
    TitledBorder naslovSlider_alfa0 = new TitledBorder("Početni kut gibanja tijela");
    naslovSlider_alfa0.setTitleFont(new Font("Tahoma",Font.BOLD,10));
    Slider_alfa0.setToolTipText("Početni kut pod kojim se tijelo počinje gibati
(ukoliko je brzina veća od nule). Ovaj kut se računa od pozitivne x-osi.");
    Slider_alfa0.setBorder(naslovSlider_alfa0);
    Slider_alfa0.setFont(new Font("Tahoma",Font.BOLD,9));
    Slider_alfa0.setMinorTickSpacing(5);
    Slider_alfa0.setPaintLabels(true);
    Slider_alfa0.setPaintTicks(true);
    Slider_alfa0.setMajorTickSpacing(45);
    Slider_alfa0.setMaximum(360);
    Slider_alfa0.setValue((int)Math.toDegrees(alfa0));
    Slider_alfa0.addChangeListener(new ChangeListener() {
      public void stateChanged(ChangeEvent changeevent)
      {
```

```
    Promijeni_alfa0(changeevent);
  }
});
prvipanel.add(Slider_alfa0,izgledpanela);


//JEDANAESTI OBJEKT - TEKSTNO POLJE "POCETNI KUT GIBANJA TIJELA"
izgledpanela.gridx = 1;
izgledpanela.gridy = 7;
izgledpanela.insets = new Insets(5,5,5,5);
Tekstpolje_alfa0 = new JTextField();
Tekstpolje_alfa0.setEnabled(false);
Tekstpolje_alfa0.setColumns(1);
Tekstpolje_alfa0.setToolTipText("Pritisni ENTER nakon upisanog broja.");
Tekstpolje_alfa0.setText(String.valueOf((int)Math.toDegrees(alfa0)));
Tekstpolje_alfa0.setFont(new Font("Tahoma",Font.BOLD,20));
Tekstpolje_alfa0.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent actionevent)
  {
    Tekstpolje_alfa0ActionPerformed(actionevent);
  }
});
prvipanel.add(Tekstpolje_alfa0,izgledpanela);


//DVANAESTI OBJEKT - POMICNA SKALA "KUTNA BRZINA"
izgledpanela.gridx = 0;
izgledpanela.gridy = 8;
izgledpanela.insets = new Insets(5,5,5,5);
Slider_omega = new JSlider(JSlider.HORIZONTAL,OMEGA_MIN,OMEGA_MAX,OMEGA_INIT);
Slider_omega.setEnabled(false);
TitledBorder naslovSlider_omega = new TitledBorder("Iznos kutne brzine (rad/s)");
naslovSlider_omega.setTitleFont(new Font("Tahoma",Font.BOLD,10));
Slider_omega.setToolTipText("Određuje se kutna brzina rotiranja sustava.");
Slider_omega.setBorder(naslovSlider_omega);
Slider_omega.setFont(new Font("Tahoma",Font.BOLD,9));
Slider_omega.setMinorTickSpacing(10);
Slider_omega.setMajorTickSpacing(50);
Slider_omega.setPaintTicks(true);
Slider_omega.addChangeListener(new ChangeListener() {
  public void stateChanged(ChangeEvent changeevent)
  {
    Promijeni_omega(changeevent);
  }
});
Hashtable<Integer, JLabel> labelTable = new Hashtable<Integer, JLabel>();
labelTable.put(new Integer(100), new JLabel("0.1"));
labelTable.put(new Integer(200), new JLabel("0.2"));
labelTable.put(new Integer(300), new JLabel("0.3"));
Slider_omega.setLabelTable(labelTable);
Slider_omega.setPaintLabels(true);
prvipanel.add(Slider_omega,izgledpanela);


//TRINAESTI OBJEKT - TEKSTNO POLJE "KUTNA BRZINA"
izgledpanela.gridx = 1;
```

```
izgledpanela.gridy = 8;
izgledpanela.insets = new Insets(5,5,5,5);
Tekstpolje_omega = new JTextField();
Tekstpolje_omega.setEnabled(false);
Tekstpolje_omega.setColumns(1);
Tekstpolje_omega.setToolTipText("Pritisni ENTER nakon upisanog broja.");
Tekstpolje_omega.setText(String.valueOf((double)OMEGA_INIT/1000));
Tekstpolje_omega.setFont(new Font("Tahoma",Font.BOLD,18));
prvipanel.add(Tekstpolje_omega,izgledpanela);
Tekstpolje_omega.addActionListener(new ActionListener() {
  public void actionPerformed(ActionEvent actionevent)
  {
    Tekstpolje_omegaActionPerformed(actionevent);
  }
});


//CETRNAESTI OBJEKT - NAZIV "Napomena - kutna brzina"
izgledpanela.insets = new Insets(5,5,5,5);
izgledpanela.gridx = 0;
izgledpanela.gridy = 9;
napomena = new JLabel("Kutna brzina ima smjer iz ekrana prema van!");
napomena.setForeground(Color.BLACK);
napomena.setFont(new Font("Tahoma",Font.BOLD,10));
prvipanel.add(napomena,izgledpanela);



//PETNAESTI OBJEKT - GUMB "PROMIJENI POCETNO STANJE"
izgledpanela.gridx = 0;
izgledpanela.gridy = 10;
izgledpanela.gridwidth=2;
izgledpanela.insets = new Insets(5,5,5,5);
promijeni_poc_stanje = new JButton("Promijeni početno stanje");
promijeni_poc_stanje.addActionListener(this);
promijeni_poc_stanje.setEnabled(true);
prvipanel.add(promijeni_poc_stanje,izgledpanela);


//SESTNAESTI OBJEKT - GUMB "POSTAVI"
izgledpanela.gridx = 0;
izgledpanela.gridy = 11;
izgledpanela.insets = new Insets(5,5,5,5);
postavi_stanje = new JButton("Postavi");
postavi_stanje.addActionListener(this);
postavi_stanje.setEnabled(false);
prvipanel.add(postavi_stanje,izgledpanela);


//SEDAMNAESTI OBJEKT - NAZIV "K. Tuk, 2007."
izgledpanela.insets = new Insets(5,5,5,5);
izgledpanela.gridx = 0;
izgledpanela.gridy = 12;
izgledpanela.weighty= 1.0;
izgledpanela.anchor = GridBagConstraints.SOUTH;
izgledpanela.fill = GridBagConstraints.NONE;
```

```
  autor = new JLabel("K. Tuk, 2007.");
  autor.setForeground(Color.BLACK);
  autor.setFont(new Font("Tahoma",Font.BOLD,12));
  prvipanel.add(autor,izgledpanela);

  //prvipanel (na lijevoj strani appleta), postava na applet
  add(prvipanel,izgledappleta);


  //postavke polozaja za drugi objekt na appletu - drugipanel
  izgledappleta.gridx=1;
  izgledappleta.gridy=0;
  izgledappleta.weightx =4.0;
  izgledappleta.weighty=1.0;
  izgledappleta.fill=GridBagConstraints.BOTH;


  //DRUGIPANEL (na desnoj strani appleta), postava na applet
  drugipanel = new crtacipanel();
  drugipanel.setBackground(new Color(220,220,220));
  add(drugipanel,izgledappleta);


  //dretva vazna za animaciju
  dretva = new Thread(this);
  dretva.start();
}

//metoda RUN koja sluzi za animaciju
public void run(){
  long tm = System.currentTimeMillis();
  while(Thread.currentThread()== dretva){
    drugipanel.repaint();
    tm = tm + delay ;
    try{
      Thread.sleep(Math.max(0, tm - System.currentTimeMillis()));
    }
    catch(InterruptedException e){
      System.err.println("Greska u run() metodi");
    }
  }
}

//metoda PROMIJENI odgovorna za izracunavanje putanje tijela pod utjecajem
//centrifugalne i coriolisove sile
private void promijeni(){

  X1 = X0 + Vx0 * deltat; Y1 = Y0 + Vy0 * deltat;
  r1 = Math.sqrt(X1*X1 + Y1*Y1);
  Vx1 = Vx0 + Ax0 * deltat; Vy1 = Vy0 + Ay0 * deltat;
  V1 = Math.sqrt(Vx1*Vx1 + Vy1*Vy1);
  temp = Math.atan2(Vy1,Vx1) + Math.toRadians(270);
  fi1 = Math.atan2(Y1,X1);
  Xcor = V1 * omega * Math.cos(temp) * 5;
  Ycor = V1 * omega * Math.sin(temp) * 5;
  Xcf = (omega*omega)*r1*Math.cos(fi1) * 5;
```

```
    Ycf = (omega*omega)*r1*Math.sin(fi1) * 5;
    Ax1=V1 * omega * Math.cos(temp) + (omega*omega)*r1*Math.cos(fi1);
    Ay1=V1 * omega * Math.sin(temp) + (omega*omega)*r1*Math.sin(fi1);

    X0 = X1; Y0 = Y1; Vx0 = Vx1; Vy0 = Vy1; Ax0 = Ax1; Ay0 = Ay1;

    //pohranjivanje nove tocke putanje, koja ce sluziti za crtanje putanje
    putanja.add(new Coord(X0, Y0));
    brojac +=1;
  }

  //metoda STRELICA koja izracunava i crta dvije linije koje predstavljaju strelicu
  private void strelica(Arc2D.Double ovacrta){

      double poc_strX = ovacrta.getStartPoint().getX();
      double poc_strY = ovacrta.getStartPoint().getY();
      double kraj_strX = ovacrta.getEndPoint().getX();
      double kraj_strY = ovacrta.getEndPoint().getY();
      double beta = Math.atan2(kraj_strY-poc_strY,kraj_strX-poc_strX);
      double poc_s1X = 20*Math.cos(beta+Math.toRadians(30));
      double poc_s1Y = 20*Math.sin(beta+Math.toRadians(30));
      double poc_s2X = 20*Math.cos(beta-Math.toRadians(20));
      double poc_s2Y = 20*Math.sin(beta-Math.toRadians(20));
      Line2D.Double poc_str1 = new Line2D.Double(poc_strX,poc_strY,
poc_s1X+poc_strX,poc_s1Y+poc_strY);
      Line2D.Double poc_str2 = new Line2D.Double(poc_strX,poc_strY,
poc_s2X+poc_strX,poc_s2Y+poc_strY);
      g2.draw(poc_str1);
      g2.draw(poc_str2);
  }

  //metoda STRELICA_VEKT koja izracunava i crta dvije linije koje predstavljaju
  //strelicu (sa drugim postavkama)
  private void strelica_vekt(Line2D.Double ovacrta){

      double poc_strX = ovacrta.getX1();
      double poc_strY = ovacrta.getY1();
      double kraj_strX = ovacrta.getX2();
      double kraj_strY = ovacrta.getY2();
      double beta = Math.atan2(kraj_strY-poc_strY,kraj_strX-poc_strX);
      double poc_s1X = 20*Math.cos(beta+Math.toRadians(20));
      double poc_s1Y = 20*Math.sin(beta+Math.toRadians(20));
      double poc_s2X = 20*Math.cos(beta-Math.toRadians(20));
      double poc_s2Y = 20*Math.sin(beta-Math.toRadians(20));
      Line2D.Double poc_str1 = new Line2D.Double(poc_strX,poc_strY,
poc_s1X+poc_strX,poc_s1Y+poc_strY);
      Line2D.Double poc_str2 = new Line2D.Double(poc_strX,poc_strY,
poc_s2X+poc_strX,poc_s2Y+poc_strY);
      g2.draw(poc_str1);
      g2.draw(poc_str2);
  }

  //metoda STOP
  public void stop(){
    dretva=null;
  }
```

```
  //metoda DESTROY koja se zadnja poziva, oslobadjaju se resursi
  public void destroy(){
  }

  //unutrasnja klasa CRTACIPANEL koja sadrzi metodu PAINT koja sluzi za crtanje
  class crtacipanel extends JPanel{
    public void paint(Graphics g){

      g2 = (Graphics2D) g;
      int width = getSize().width;
      int height = getSize().height;

      g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING,
RenderingHints.VALUE_ANTIALIAS_ON);

      AffineTransform at = new AffineTransform();
      at.translate(width/2,height/2);
      g2.transform(at);
      at.setToScale(1.0,-1.0);
      //postavlja se orijentacija koord. sustava: x-os je usmjerena desno,
      //a y-os prema gore
      g2.transform(at);

      //podloga
      g2.setColor(new Color(220,220,220));
      Rectangle2D.Double podloga = new Rectangle2D.Double(0-width/2,-100-width/2,
width,width+195);
      g2.fill(podloga);

      //postavlja se puni krug koji predstavlja svemirski kotac, vrtuljak
      g2.setColor(new Color(16,222,217));
      Ellipse2D.Double elipsa = new Ellipse2D.Double(0-r,0-r,2*r,2*r);
      g2.fill(elipsa);

      //strelice koje oznacuju smjer vrtnje
      g2.setColor(new Color(89,89,89));
      g2.setStroke(new BasicStroke(2));
      Arc2D.Double smjerDD = new Arc2D.Double(0-rSMJ,0-rSMJ,2*rSMJ,2*rSMJ,35,15,0);
      g2.draw(smjerDD);
      strelica(smjerDD);
      Arc2D.Double smjerDL = new Arc2D.Double(0-rSMJ,0-rSMJ,2*rSMJ,2*rSMJ,125,15,0);
      g2.draw(smjerDL);
      strelica(smjerDL);
      Arc2D.Double smjerGL = new Arc2D.Double(0-rSMJ,0-rSMJ,2*rSMJ,2*rSMJ,215,15,0);
      g2.draw(smjerGL);
      strelica(smjerGL);
      Arc2D.Double smjerGD = new Arc2D.Double(0-rSMJ,0-rSMJ,2*rSMJ,2*rSMJ,305,15,0);
      g2.draw(smjerGD);
      strelica(smjerGD);
      g2.setStroke(new BasicStroke(1));

      //x i y os na punom krugu, radi boljeg uvida
      g2.setColor(new Color(0,0,0));
      Line2D.Double xos = new Line2D.Double(0,r,0,-r);
      Line2D.Double yos = new Line2D.Double(r,0,-r,0);
```

```
      g2.draw(xos);
      g2.draw(yos);


      //PUTANJA TIJELA
      g2.setColor(new Color(241,0,0));
      if((X0==0)&&(Y0==0)&&(Vx0==0)&&(Vy0==0)){}
      //u slucaju da je polozaj tijela u sredistu vrtnje, a njegova brzina 0,
tijelo ostaje u sredistu vrtnje,
      //pa da se izbjegne nepotrebno racunanje, koristi se ovo grananje
      else
      {
      //uz zadovoljeni uvjet, ova petlja crta putanju tijela
      for(int i = 0; i < putanja.size(); i++)
        {
          g2.fill(new Ellipse2D.Double(((Coord)putanja.get(i)).x - rPUT,
((Coord)putanja.get(i)).y - rPUT,2*rPUT,2*rPUT));
        }
      }

      //crtanje tijela na punom krugu, odredjen polozajem X0 i Y0
      g2.setColor(new Color(15,101,41));
      Ellipse2D.Double elli = new Ellipse2D.Double(X0-rTIJ,Y0-rTIJ,2*rTIJ,2*rTIJ);
      g2.fill(elli);


      if (centrif.isSelected()==true) {
        //CRTANJE CENTRIFUGALNE SILE
        if((X0==0)&&(Y0==0)){}
        //ako se tijelo nalazi u sredistu vrtnje, centrifugalna sila je nula,
        //pa ju nije potrebno crtati
        else{
          g2.setColor(new Color(255,255,0));
          Line2D.Double centrifsila = new Line2D.Double(Xcf+X0,Ycf+Y0,X0,Y0);
          g2.draw(centrifsila);
          strelica_vekt(centrifsila);
        }
      }

      if (coriol.isSelected()==true) {
        //CRTANJE CORIOLISOVE SILE
        if((Vx0==0)&&(Vy0==0)){}
        //ako je pocetna brzina tijela nula, nije potrebno crtati coriolisovu silu
        else{
          g2.setColor(new Color(0,0,212));
          g2.setStroke(new BasicStroke(1));
          Line2D.Double coriolissila = new Line2D.Double(Xcor+X0,Ycor+Y0,X0,Y0);
          g2.draw(coriolissila);
          strelica_vekt(coriolissila);
        }
      }


      if (brzina.isSelected()==true) {
        //CRTANJE VEKTORA BRZINE
        if((Vx0==0)&&(Vy0==0)){}
```

```
      //ako je pocetna brzina tijela nula, nije potrebno crtati strelicu
      //koja predstavlja vektor brzine
      else{
        g2.setColor(new Color(225,101,41));
        g2.setStroke(new BasicStroke(2));
        Line2D.Double brzina = new Line2D.Double(Vx0+X0,Vy0+Y0,X0,Y0);
        g2.draw(brzina);
        strelica_vekt(brzina);
        g2.setStroke(new BasicStroke(1));
      }
    }

    //kada se pritisne gumb za pocetak simulacije, potrebno je obaviti
    //racunanje putanje, brzine, sila
    if(checkPaint == true){
      pauza.setEnabled(true);
      pocni.setEnabled(false);

      if((X0==0)&&(Y0==0)&&(Vx0==0)&&(Vy0==0)){}
      else
      {
        promijeni();
      }
    }

    //tijelo se u simulaciji giba sve dok ne dodje do ruba punog kruga,
    //tako prestaje crtanje
    if(X0*X0+Y0*Y0 > r*r){
      pocni.setEnabled(false);
      pauza.setEnabled(false);
      checkPaint = false;
    }

    //ponovno crtanje
    repaint();
  }
}

//metoda koja sadrzi razlicite slucajeve dogadjaja - kada se pritisne koji gumb
public void actionPerformed(ActionEvent e) {
  if(e.getSource() == pocni){
    promijeni_poc_stanje.setEnabled(false);
    checkPaint = true;
    pocetno_stanje.setEnabled(true);
  }
  if(e.getSource() == pauza){
    checkPaint = false;
    pauza.setEnabled(false);
    pocni.setText("Nastavi");
    pocni.setEnabled(true);
  }
  if(e.getSource()== pocetno_stanje){
    pocni.setText("Kreni");
    pocni.setEnabled(true);
    pauza.setEnabled(false);
    promijeni_poc_stanje.setEnabled(true);
```

```
      putanja.clear();
      resetiraj();
      repaint();
    }
    if(e.getSource() == promijeni_poc_stanje){
      pocni.setEnabled(false);
      pauza.setEnabled(false);
      pocetno_stanje.setEnabled(false);
      promijeni_poc_stanje.setEnabled(false);
      postavi_stanje.setEnabled(true);
      Slider_r0.setEnabled(true);
      Slider_fi0.setEnabled(true);
      Slider_V0.setEnabled(true);
      Slider_alfa0.setEnabled(true);
      Slider_omega.setEnabled(true);
      Tekstpolje_alfa0.setEnabled(true);
      Tekstpolje_fi0.setEnabled(true);
      Tekstpolje_V0.setEnabled(true);
      Tekstpolje_r0.setEnabled(true);
      Tekstpolje_omega.setEnabled(true);
    }
    if(e.getSource() == postavi_stanje){
      pocni.setEnabled(true);
      pauza.setEnabled(false);
      promijeni_poc_stanje.setEnabled(false);
      pocetno_stanje.setEnabled(false);
      postavi_stanje.setEnabled(false);
      Slider_r0.setEnabled(false);
      Slider_fi0.setEnabled(false);
      Slider_V0.setEnabled(false);
      Slider_alfa0.setEnabled(false);
      Slider_omega.setEnabled(false);
      Tekstpolje_alfa0.setEnabled(false);
      Tekstpolje_fi0.setEnabled(false);
      Tekstpolje_V0.setEnabled(false);
      Tekstpolje_r0.setEnabled(false);
      Tekstpolje_omega.setEnabled(false);
      repaint();
    }
  }

  //metoda kojom se mijenja vrijednost r0 preko pomicne skale
  private void Promijeni_r0(ChangeEvent changeevent)
  {
    r0=((JSlider)changeevent.getSource()).getValue();
    Tekstpolje_r0.setText(String.valueOf
(((JSlider)changeevent.getSource()).getValue()));
    putanja.clear();
    resetiraj();
  }

  //metoda kojom se ostvaruje upisivanje vrijednosti u R0 tekstno polje
  private void Tekstpolje_r0ActionPerformed(ActionEvent actionevent)
  {
    try
    {
```

```
      Slider_r0.setValue((int)Double.parseDouble
(((JTextField)actionevent.getSource()).getText()));
    }
    catch(NumberFormatException _ex)
    {
      int pomocni = (int)POCRADIJUS;
      Slider_r0.setValue(pomocni);
      Tekstpolje_r0.setText(String.valueOf(pomocni));
    }
  }

  //metoda kojom se mijenja vrijednost fi0 preko pomicne skale
  private void Promijeni_fi0(ChangeEvent changeevent)
  {
    fi0=Math.toRadians(((JSlider)changeevent.getSource()).getValue());
    Tekstpolje_fi0.setText(String.valueOf
(((JSlider)changeevent.getSource()).getValue()));
    putanja.clear();
    resetiraj();
  }

  //metoda kojom se ostvaruje upisivanje vrijednosti u FI0 tekstno polje
  private void Tekstpolje_fi0ActionPerformed(ActionEvent actionevent)
  {
    try
    {
      Slider_fi0.setValue((int)Double.parseDouble
(((JTextField)actionevent.getSource()).getText()));
    }
    catch(NumberFormatException _ex)
    {
      Slider_fi0.setValue((int)Math.toDegrees(POCFI));
      Tekstpolje_fi0.setText(String.valueOf((int)Math.toDegrees(POCFI)));
    }
  }

  //metoda kojom se mijenja vrijednost V0 preko pomicne skale
  private void Promijeni_V0(ChangeEvent changeevent)
  {
    V0=((JSlider)changeevent.getSource()).getValue();
    Tekstpolje_V0.setText(String.valueOf
(((JSlider)changeevent.getSource()).getValue()));
    putanja.clear();
    resetiraj();
  }

  //metoda kojom se ostvaruje upisivanje vrijednosti u V0 tekstno polje
  private void Tekstpolje_V0ActionPerformed(ActionEvent actionevent)
  {
    try
    {
      Slider_V0.setValue((int)Double.parseDouble
(((JTextField)actionevent.getSource()).getText()));
    }
    catch(NumberFormatException _ex)
    {
```

```
      int pomocni = (int)POCBRZINA;
      Slider_V0.setValue(pomocni);
      Tekstpolje_V0.setText(String.valueOf(pomocni));
    }
  }

  //metoda kojom se mijenja vrijednost ALFA0 preko pomicne skale
  private void Promijeni_alfa0(ChangeEvent changeevent)
  {
    alfa0=Math.toRadians(((JSlider)changeevent.getSource()).getValue());
    Tekstpolje_alfa0.setText(String.valueOf
(((JSlider)changeevent.getSource()).getValue()));
    putanja.clear();
    resetiraj();
  }

  //metoda kojom se ostvaruje upisivanje vrijednosti u ALFA0 tekstno polje
  private void Tekstpolje_alfa0ActionPerformed(ActionEvent actionevent)
  {
    try
    {
      Slider_alfa0.setValue((int)Double.parseDouble
(((JTextField)actionevent.getSource()).getText()));
    }
    catch(NumberFormatException _ex)
    {
      Slider_alfa0.setValue((int)Math.toDegrees(POCALFA));
      Tekstpolje_alfa0.setText(String.valueOf((int)Math.toDegrees(POCALFA)));
    }
  }

  //metoda kojom se mijenja vrijednost OMEGA preko pomicne skale
  private void Promijeni_omega(ChangeEvent changeevent)
  {
    double pomocni =((JSlider)changeevent.getSource()).getValue();
    omega = pomocni / 1000;
    Tekstpolje_omega.setText(String.valueOf(omega));
    putanja.clear();
    resetiraj();
  }

  //metoda kojom se ostvaruje upisivanje vrijednosti u OMEGA tekstno polje
  private void Tekstpolje_omegaActionPerformed(ActionEvent actionevent)
  {
    try
    {
      Slider_omega.setValue((int)(1000*Double.parseDouble
(((JTextField)actionevent.getSource()).getText())));
    }
    catch(NumberFormatException _ex)
    {
      int pomocni = (int)(1000*omega0);
      Slider_omega.setValue(pomocni);
      Tekstpolje_omega.setText(String.valueOf((double)pomocni/1000));
    }
  }
```

```
  //metoda RESETIRAJ koja sluzi da postavi vrijednosti na pocetak (osim onih
  //vrijednosti koje se mogu mijenjati)
  public void resetiraj(){
    brojac = 0;
    checkPaint = false;

    X0 = r0 * Math.cos(fi0); Y0 = r0 * Math.sin(fi0);
    Vx0 = V0 * Math.cos(alfa0); Vy0 = V0 * Math.sin(alfa0);
    temp = Math.atan2(Vy0,Vx0) + Math.toRadians(270);
    Xcor = V0 * omega * Math.cos(temp) * 5;
    Ycor = V0 * omega * Math.sin(temp) * 5;
    Xcf = (omega*omega)*r0*Math.cos(fi0) * 5;
    Ycf = (omega*omega)*r0*Math.sin(fi0) * 5;
    Ax0=V0 * omega * Math.cos(temp) + (omega*omega)*r0*Math.cos(fi0);
    Ay0=V0 * omega * Math.sin(temp) + (omega*omega)*r0*Math.sin(fi0);
    putanja.add(new Coord(X0, Y0));

  }
}

//klasa COORD koja sluzi za tocke putanje
class Coord
{
    double x;
    double y;

    Coord(double xx, double yy)
    {
        x = xx;
        y = yy;
    }
}
```