

Udhëzuesi i Kturtle

**Cies Breijs, Anne-Marie Mahfouf, Mauricio Piacentini, dhe
Dashamir Hoxha**



Udhëzuesi i Kturtle

Përmbajta

1 Hyrje	1
1.1 Çfarë është TurtleScript?	1
1.2 Veçoritë e Kturtle	1
2 Përdorimi i Kturtle	3
2.1 Shkrimorja	3
2.2 Hapësira e Vizatimit	4
2.3 Shqyrtuesi	4
2.4 Rreshti i Mjeteve	4
2.5 Rreshti i Menuve	4
2.5.1 Menuja Skedar	4
2.5.2 Menuja Shkrim	5
2.5.3 Menuja Tabela	6
2.5.4 Menuja Nis	6
2.5.5 Menuja Mjetet	7
2.5.6 Menuja Rregullimet	7
2.5.7 Menuja Ndihmë	8
2.6 Shiriti i Gjendjes	8
3 Hapat e Para	9
3.1 Hapat e para me TurtleScript: takoni Breshkën!	9
3.1.1 Lëvizjet e Breshkës	10
3.1.2 Shembuj të tjerë	10
4 Referenca e Programimit për TurtleScript	12
4.1 Gramatika e TurtleScript	12
4.1.1 Komentet	12
4.1.2 Komandat	13
4.1.3 Numrat	13
4.1.4 Vargjet	13
4.1.5 Vlerat e vërtetësisë (e vërtetë / e rreme)	14

4.2	Veprimet matematikore dhe krahasuese	14
4.2.1	Veprimet matematike	14
4.2.2	Veprimet logjike të vërtetësisë (e vërtetë / e rreme)	15
4.2.2.1	Disa shembuj më të përparuar	15
4.2.3	Veprimet krahasuese	16
4.3	Komandat	16
4.3.1	Lëvizja e breshkës	16
4.3.2	Ku është breshka?	18
4.3.3	Breshka ka një laps	18
4.3.4	Komandat për të kontrolluar tabelën e vizatimit	19
4.3.5	Komandat për të pastruar	19
4.3.6	Breshka është një figurinë	19
4.3.7	A mund të shkruaj breshka?	20
4.3.8	Komandat matematikore	20
4.3.9	Bashkëbisedimi me programin	22
4.4	Vlerëdhënia e ndryshoreve	22
4.5	Kontrollimi i zbatimit	23
4.5.1	Bëjeni breshkën të presë	23
4.5.2	Zbato “nëse”	24
4.5.3	Nëse jo, ose me fjalë të tjera: “përndryshe”	24
4.5.4	Cikli “përderisa”	24
4.5.5	Cikli “përsërit”	25
4.5.6	Cikli “për_çdo”, një cikël numëruar	25
4.5.7	Braktisja e një cikli	25
4.5.8	Ndaloni zbatimin e programit	26
4.6	Krijoni komandat tuaja me ‘mëso’	26
5	Fjalor	28
6	Udhëzuesi i Përkthyesit të KTurtle	32
7	Vlerësimet dhe Liçenca	33
A	Instalimi	34
A.1	Ku ta gjejmë KTurtle	34
A.2	Përpunimi dhe Instalimi	34

Tabelat

4.2	Llojet e pyetjeve	16
5.2	Llojet e ndryshme të kodit dhe ngjyrat e theksimit të tyre	30
5.4	Kombinime RGB që përdoren shpesh	31

Përmbledhja

KTurtle është një mjedis programimi shkollor që ka për qëllim ta bëjë sa më të lehtë mësimin e programimit. Për ta arritur këtë KTurtle vë në dispozicion të gjitha mjetet e programimit në ndërfaqen e përdorimit. Gjuha e programimit e përdorur është TurtleScript, e cila lejon që komandat e saj të përkthehen.

Kapitulli 1

Hyrje

KTurtle është një mjedis programimi arsimor që përdor [TurtleScript](#), një gjuhë programimi kryesisht e bazuar dhe e frymëzuar nga Logo. Qëllimi i KTurtle është që ta bëjë programimin sa më të lehtë dhe të arritshëm që të jetë e mundur. Kjo e bën KTurtle të përshtatshëm për tu mësuar fëmijëve bazat e matematikës, gjeometrisë dhe ... programimit. Një nga karakteristikat kryesore të TurtleScript është mundësia për ti përkthyer komandat në gjuhën e folur të programuesit.

KTurtle e ka marrë emrin nga ‘breshka’ që luan një rol kryesor në mjedisin e programimit. Studenti zakonisht do të udhëzojë breshkën, duke përdorur komandat e TurtleScript, që të bëjë një vizatim mbi [tabelë](#).

1.1 Çfarë është TurtleScript?

TurtleScript, gjuha e programimit e përdorur në KTurtle, është frymëzuar nga familja e gjuhëve të programimit Logo. Versioni i parë i gjuhës Logo u krijua nga Seymour Papert në MIT Artificial Intelligence Laboratory (Laboratori i Inteligjencës Artificiale i MIT) në 1967 si një degë e gjuhës së programimit LISP. Prej atëherë shumë versione të gjuhës Logo kanë dalë. Rreth vitit 1980 Logo po merte hov, me versionet për sistemet MSX, Commodore, Atari, Apple II dhe IBM PC. Këto versione kanë qenë kryesisht për qëllime edukative. MIT ende mirëmban një [faqe interneti për Logo](#) e cila ka një listë të disa prej ndërtimeve më popullore të gjuhës.

TurtleScript ka një tipar që gjendet në shumë mishërime të tjera të gjuhës Logo: aftësinë për ti përkthyer komandat që ti përshtaten gjuhës amtare të studentit. Kjo veçori e bën më të lehtë fillimin për nxënësit që kanë pak ose aspak njohuri të gjuhës angleze. Përveç këtij tipari, KTurtle ka edhe [shumë veçori të tjera](#) që kanë për qëllim lehtësimin e përvojës fillestare të nxënësve me programimin.

1.2 Veçoritë e KTurtle

KTurtle ka tipare të këndshme që e bëjnë fillimin e programimit si një llokume. Disa nga tiparet kryesore janë theksuar më poshtë:

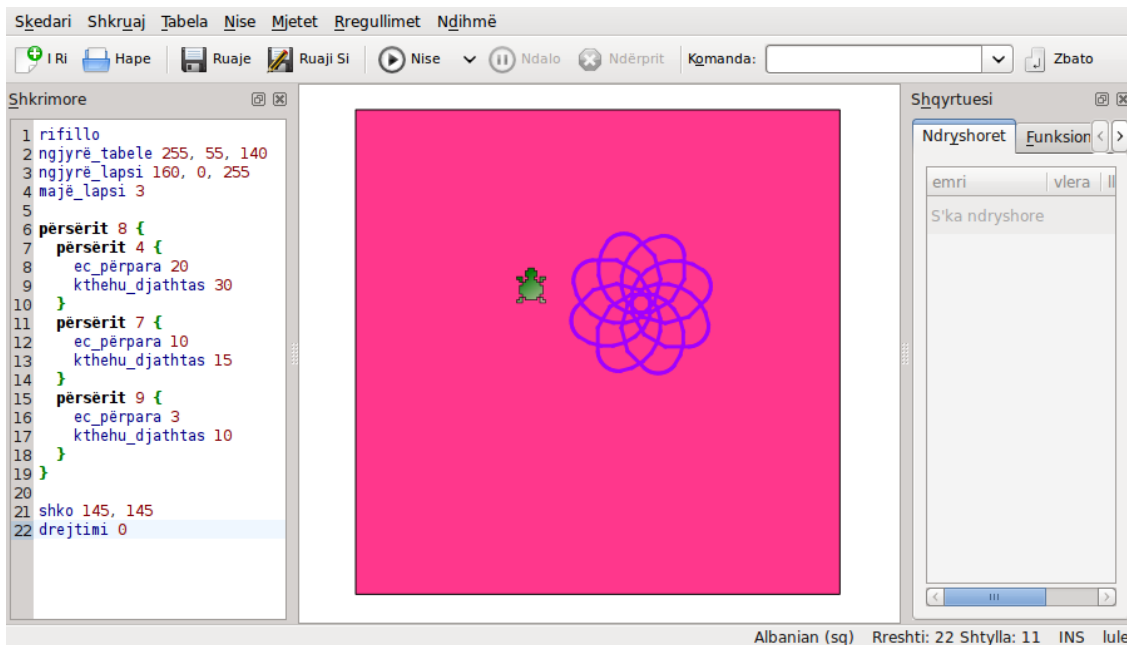
- Një mjedis i ndërthurur me interpretuesin TurtleScript, [shkrimoren](#), [tabelën e vizatimit](#) dhe mjete të tjera, të gjitha në një aplikim të vetëm (pa varësi të tepërta).
- Mundësia për të përkthyer komandat e TurtleScript-it, duke përdorur strukturën përkthyesë të KDE-së.

Udhëzuesi i KTurtle

- TurtleScript ka mbështetje për funksionet e përcaktuara nga përdoruesi, për rekursivitetin dhe për ndërrimin dinamik të tipit.
- Zbatimi i programit mund të ngadalësohet, pezullohet ose ndalohet në çdo kohë.
- Një [shkrimore](#) e fuqishme, me theksues sintakse intuitiv, numërim rreshtash, shënues gabimesh, zbatim vizual, e të tjera.
- [Tabela](#) ku breshka vizaton, mund të shtypet në letër ose të ruhet ose si një imazh (PNG) ose si një vizatim (SVG).
- Ndhma rrethore, aty ku keni nevojë për të. Vetëm shtypni F2 (ose shiko **Ndihmë** → **Ndihmë për: ...**) për të hapur ndihmën për atë pjesë të programit ku ndodhet kursori.
- Një dritare gabimesh që lidh mesazhet e gabimit me gabimet në program dhe i shënon ato me të kuqe.
- Terminologji programimi e thjeshtuar.
- Shembuj programesh të gatshme, për ta bërë të lehtë fillimin. Këta shembuj përkthehen duke përdorur strukturën përkthyesë të KDE.

Kapitulli 2

Përdorimi i KTurtle



Dritarja kryesore e KTurtle ka tre pjesë kryesore: **shkrimoren** (1) në të majtë ku ju shkruani programin TurtleScript, **tabelën e vizatimit** (2) në të djathtë ku breshka bën vizatimin, dhe **shqyrtuesin** (3) i cili u jep informata kur programi është në punë. Përveç këtyre është edhe **shiriti i menyve** (5) ku mund të gjenden të gjitha veprimet, **shiriti i mjeteve** (4) që ju lejon të zgjidhni shpejt veprimet më të përdorshme, **vendi i komandave**, që ju mund ta përdorni për të provuar një komandë, dhe **shiriti i gjendjes** (përgjatë pjesës së poshtme të dritares), ku mund të gjeni informata mbi gjendjen e KTurtle.

2.1 Shkrimorja

Në shkrimore ju shkruani programin TurtleScript. Shumica e funksioneve të shkrimores mund të gjenden në menutë **Skedarët** dhe **Redaktim**. Shkrimorja mund të ngjitet në secilën anë të dritares kryesore ose mund të shkëputet dhe të vendoset kudo në ekran.

Ka mënyra të ndryshme për të vendosur ca komanda në shkrimore. Mënyra më e lehtë është të merrni një shembull: zgjidhni **Skedar** → **Shembuj** në **menunë Skedar** dhe përzgjidhni një

shembull. Skedari shembull që zgjidhni do të hapet në [shkriomore](#), dhe atëherë po të doni ju mund të përdorni **Nis** → **Nis** te menutë ose **Nis** te mjetet për të filluar zbatimin e programit.

Ju mund të hapni skedarë TurtleScript duke përdorur **Skedar** → **Hap...**

Mënyra e tretë është që ta shkruani direkt programin tuaj në shkriomore ose të kopjoni disa rreshta prej diku tjetër.

2.2 Hapësira e Vizatimit

Hapësira (ose tabela) e vizatimit është vendi ku rri breshka. Aty breshka vizaton sipas komandave që merr. Pasi të vendosni komanda në [Shkriomore](#) dhe të nisni zbatimin e tyre, dy gjëra mund të ndodhin: ose komandat zbatohen për bukuri, dhe ka shumë të ngjarë të shihni ndryshime në tabelë, ose ka ndonjë gabim në rreshtat e programit, dhe në këtë rast do të shfaqet kartela e gabimeve që shpjegon se çfarë gabimi ka ndodhur.

Ju mund ta ta zvogëloni ose ta zmadhoni tabelën e vizatimit duke përdorur rrotëzën e miushit.

2.3 Shqyrtuesi

Shqyrtuesi ju informon për ndryshoret, komandat e mësuara dhe tregon degëzimin e komandave gjatë kohës që programi është në punë.

Shqyrtuesi mund të vendoset në secilën anë të dritares kryesore ose mund të shkëputet dhe të vendoset kudo në ekran.

2.4 Rreshti i Mjeteve

Këtu ju mund të arrini shpejt veprimet më të përdorshme. Te mjetet gjendet gjithashtu edhe **Komanda** ku ju mund të jepni shpejt një komandë; kjo mund të jetë e dobishëm në rast se doni të provoni një komandë pa e ndryshuar përmbajtjen e [Shkrimores](#).

Ju mund ta ndryshoni rreshtin e mjeteve duke përdorur **Rregullimet** → **Ndrysho Rreshtin e Mjeteve...** për ta përshtatur më mirë me shijet tuaja.

2.5 Rreshti i Menuve

Në rreshtin e menuve ju mund të gjeni të gjitha veprimet e mundshme në KTurtle. Ato janë në grupet e mëposhtme: **Skedar**, **Redaktim**, **Tabela**, **Nis**, **Mjetet**, **Rregullimet**, dhe **Ndihmë**. Kjo pjesë i përshkruan ato të gjitha.

2.5.1 Menuja Skedar

Skedar → **i Ri (Ctrl-R)**

Krijon një skedar të ri TurtleScript që është bosh.

Skedar → **Hap... (Ctrl-H)**

Hap një skedar TurtleScript.

Skedar → Hap të Mëparshëm

Hap një skedar TurtleScript që është hapur kohët e fundit.

Skedar → Shembuj

Hap shembuj programesh TurtleScript. Shembujt janë në gjuhën tuaj të preferuar që ju mund të zgjidhni te **Rregullimet** → **Gjuha e Programimit**.

Skedar → Merrni shembuj të tjerë...

Hap dritaren e bashkëbisedimit **Merr Gjëra të Reja Interesante** për të shkarkuar skedarë të tjerë TurtleScript, nga Interneti.

Skedar → Ruaj (Ctrl-S)

Ruan skedarin TurtleScript të hapur momentalisht.

Skedar → Ruaj Si...

Ruan skedarin TurtleScript të hapur momentalisht në një vend të caktuar.

Skedar → Ktheje në HTML...

Ruan përmbajtjen e Shkrimore në një skedar HTML që përfshin theksimin e ngjyrave.

Skedar → Shtyp... (Ctrl-P)

Shtyp në letër programin që ndodhet në shkrimore.

Skedar → Mbyll (Ctrl-Q)

Mbyll KTurtle.

2.5.2 Menuja Shkrim

Shkrim → Kthe (Ctrl-K)

Kthen mbrapsht ndryshimin e fundit në program. Numri i kthimeve në KTurtle është i pakufizuar.

Shkrim → Ribëj (Ctrl-Shift-K)

Bën përsëri një ndryshim të kthyer mbrapsht në program.

Shkrim → Merr (Ctrl-X)

Merr tekstin e përzgjedhur nga [shkrimorja](#) dhe e ruan në Kujtesë.

Shkrim → Kopjo (Ctrl-C)

Kopjon tekstin e përzgjedhur nga [shkrimorja](#) dhe e ruan në Kujtesë.

Shkrim → Vendos (Ctrl-V)

Vendos tekstin e Kujtesës në [shkrimore](#).

Shkrim → Përzgjidhe të gjithë (Ctrl-A)

Përzgjedh të gjithë tekstin nga [shkrimorja](#).

Shkrim → Gjej... (Ctrl-F)

Me këtë veprim mund të gjeni shprehje në program.

Shkrim → Gjej Tjetrën (F3)

Kjo përdoret për të gjetur vendin tjetër ku gjendet shprehja që keni kërkuar.

Shkrim → Gjej Paraardhësen (Shift-F3)

Përdoret për të gjetur vendin e mësipërm ku ndodhet shprehja e kërkuar.

Shkrim → Mënyra e Mbishkrimit (Ins)

Kalo nga mënyra e 'ndërfutjes' në të 'mbishkrimit' dhe anasjelltas.

2.5.3 Menuja Tabela

Tabela → Ruaje si Imazh (PNG)...

Ruan përmbajtjen aktuale të [tabelës](#) si një imazh i tipit PNG (Portable Network Graphics).

Tabela → Ruaj si Vizatim (SVG)...

Ruan përmbajtjen aktuale të [tabelës](#) si një imazh i tipit SVG (Scalable Vector Graphics).

Tabela → Shtyp Tabelën...

Shtyp në letër përmbajtjen aktuale të [Tabelës](#).

2.5.4 Menuja Nis

Nis → Nis (F5)

Fillon zbatimin e komandave në editor.

Ris → Pusho (F6)

Ndalon zbatimin për njëfarë kohe. Ky veprim është aktiv vetëm kur komandat janë duke u zbatuar.

Nis → Ndërprit (F7)

Ndalon zbatimin e programit. Ky veprim është aktiv vetëm kur programi është duke u zbatuar.

Nis → Shpejtësia e Lëvizjes

Paraqet një listë me shpejtësitë e mundshme të lëvizjes, e përbërë nga: **Shpejtësi Maksimale (pa ndriçuesin dhe shqyrtuesin)**, **Shpejtësi e Madhe**, **Ngadalë**, **Pak më Ngadalë**, **Shumë Ngadalë** dhe **Hap-pas-Hapi**. Kur shpejtësia e lëvizjes është **Shpejtësi e Madhe** (e parazgjedhur) zor se mund ta ndjekim se çfarë po ndodh. Ndonjëherë kështu e duam, por ndonjëherë duam të ndjekim rrjedhën e zbatimit të programit. Në këtë rast duhet ta vendosim shpejtësinë e lëvizjes **Ngadalë**, **Pak më Ngadalë** ose **Shumë Ngadalë**. Në rast se një nga mënyrat e ngadalshme është zgjedhur, pozicioni aktual i zbatimit do të tregohet në shkrimore. **Hap-pas-Hapi** do ti zbatojë komandat një e nga një.

2.5.5 Menuja Mjetet

Mjetet → Zgjedhës Drejtimi...

Hap dritaren bashkëbiseduese për të zgjedhur drejtimin.

Mjetet → Zgjedhës Ngjyre...

Hap dritaren bashkëbiseduese të zgjedhjes së ngjyrës.

2.5.6 Menuja Rregullimet

Rregullimet → Gjuha e Programimit

Zgjedh gjuhën e programit.

Rregullimet → Nxirr Shkrimoren (Ctrl-E)

Nxjerr ose heq [Shkrimoren](#).

Rregullimet → Nxirr Shqyrtuesin (Ctrl-I)

Nxjerr ose heq [Shqyrtuesin](#).

Rregullimet → Nxirr Gabimet

Nxjerr ose heq kartelën **Gabimet** e cila ka një listë të gabimeve që dalin gjatë zbatimit të programit. Nëse kjo është e aktivizuar, klikoni te **Tabela** për të parë breshkën përsëri.

Rregullimet → Shfaq Numrat e Rreshtave (F11)

Tregon numrat e rreshtave në [shkrimore](#). Kjo mund të jetë e dobishme për gjetjen e gabimeve.

Rregullimet → Shfaq Rreshtin e Mjeteve

Nxirr/Hiq rreshtin kryesor të mjeteve

Rregullimet → Shfaq Shiritin e Gjendjes

Nxirr/Hiq Shiritin e Gjendjes

Rregullimet → Konfiguro Shkurtoret...

Dritare bashkëbisedimi standarde e KDE-së për të konfiguruar shkurtoret.

Rregullimet → Konfiguro Shiritat...

Dritare bashkëbisedimi standarde e KDE-së për të konfiguruar shiritat.

2.5.7 Menuja Ndhimë

Ndhimë → Udhëzuesi i KTurtle (F1)

Hap programin e Ndhimës së KDE-së duke filluar te faqet e ndihmës për KTurtle. (ky dokument).

Ndhimë → Ç'është Kjo? (Shift+F1)

E bën shenjën e mausit si një shigjetë me një pikëpyetje. Klikimi i gjërave në KTurtle do të hapë një dritare ndihme (nëse ka ndihmë për atë gjë) që shpjegon funksionin e saj.

Ndhimë → Raporto Difekt...

Hap dritaren bashkëbiseduese për raportimin e difekteve ku mund të raportohet një difekt ose të kërkohet një përmirësim i dëshiruar.

Ndhimë → Ndërro Gjuhën e Programit...

Hap një dritare bashkëbisedimi ku mund të vendoset **Gjuha kryesore** dhe **Gjuha zëvendësuese** për këtë program.

Ndhimë → Rreth KTurtle

Shfaq të dhëna lidhur me versionin dhe autorin e programit.

Ndhimë → Rreth KDE-së

Shfaq versionin e KDE-së dhe të dhëna të tjera elementare.

Ndhimë → Ndhimë për... (F2)

Ky është një funksion shumë i dobishëm: ai siguron ndihmë për komandën në të cilën ndodhet kursori në shkrimore. P.sh. ju keni përdorur komandën *shkruaj në program*, dhe dëshironi të lexoni dhe të dini se çfarë thotë udhëzuesi për këtë komandë. Për këtë ju e lëvizni kursorin dhe e çoni te komanda *shkruaj*, pastaj shtypni F2. Udhëzuesi do të hapet tek pjesa ku janë shpjegimet mbi komandën *shkruaj*.

Kjo gjë mund të dalë e dobishme gjatë mësimit të TurtleScript.

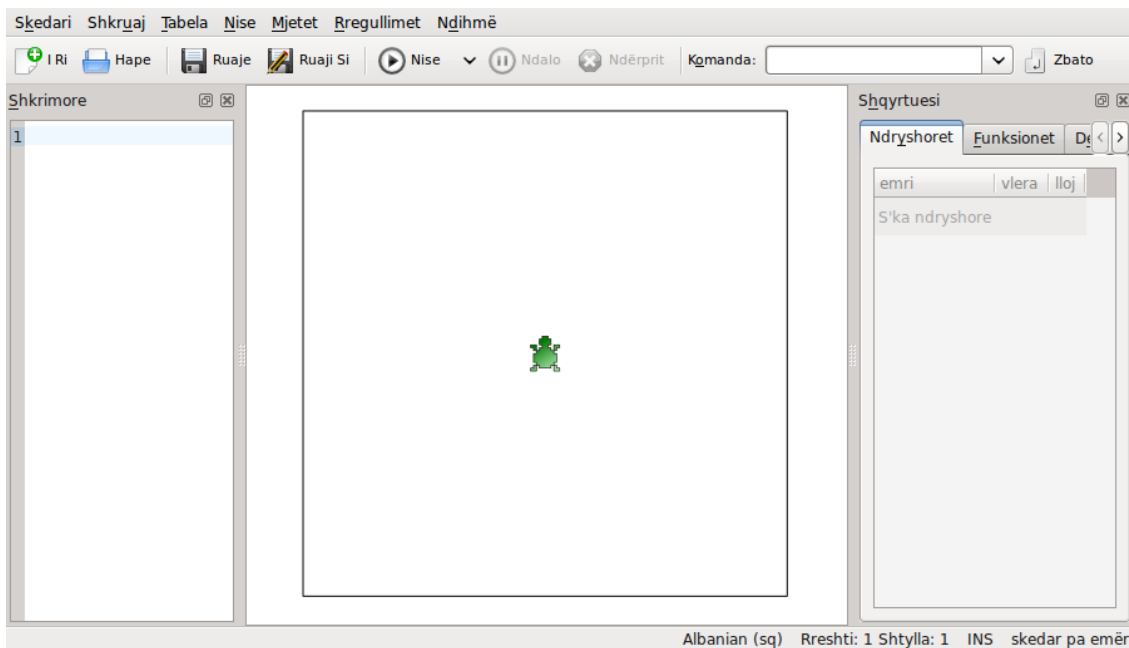
2.6 Shiriti i Gjendjes

Në shiritin e gjendjes ju mund të shihni informacione mbi gjendjen e KTurtle. Në anën e majtë ai tregon komentet mbi veprimin e fundit. Në anën e djathtë mund të gjeni pozicionin aktual të kursorit (numrat e rreshtit dhe të shtyllës). Në mes të shiritit tregohet gjuha që po përdoret për komandat.

Kapitulli 3

Hapat e Para

Kur hapni KTurtle do të shihni diçka të tillë:



Në këtë udhëzues fillestar ne supozojmë se gjuha e komandave të TurtleScript është shqip. Ju mund ta ndryshoni këtë gjuhë te **Rregullimet** → **Gjuha e Programit**. Kini parasysh se gjuha që vendoset këtu për KTurtle është ajo që përdoret për të shkruar komandat e TurtleScript, jo gjuha e përdorur nga KDE në kompjuterin tuaj dhe që përdoret për të shfaqur ndërfaqen dhe menutë e KTurtle.

3.1 Hapat e para me TurtleScript: takoni Breshkën!

Ju duhet ta keni vënë re breshkën në mes të tabelës katrore. Këtu do të mësoni si ta lëvizni atë duke përdorur komandat në shkrimore.

3.1.1 Lëvizjet e Breshkës

Le të fillojmë duke e vënë breshkën në lëvizje. Breshka jonë mund të bëjë 3 lloje lëvizjesh, (1) mund të lëvizë përpara dhe mbrapa, (2) mund të kthehet majtas dhe djathtas dhe (3) mund të shkojë (hidhet) direkt në një pozicion në ekran. Për shembull provoni këtë:

```
ec_përpara 100
kthehu_majtas 90
```

Shkruaji ose kopjoi këto komanda në shkrimore dhe zbatoi (duke përdorur [NiseNise](#)) për të parë rezultatin.

Pasi shkruat në shkrimore komandat më sipër dhe i zbatuat ato, ju mund të keni vënë re disa nga gjërat e mëposhtme:

1. Se pas zbatimit të komandave, breshka shkon lart, vizaton një vijë, dhe pastaj kthehet në të majtë. Kjo për shkak se ju keni përdorur komandat `ec_përpara` dhe `kthehu_majtas`.
2. Se ngjyra e komandave në shkrimore ndryshoi, ndërsa ju i shtypnit ato; kjo veçori quhet *theksim intuitiv* — lloje të ndryshme komandash theksohen ndryshe. Kjo e bën leximin e blloqeve të mëdha të programit më të lehtë.
3. Se breshka vizaton një vijë të hollë të zezë.
4. Ndoshta ju doli ndonjë mesazh gabimi. Kjo thjesht mund të ketë dy kuptime: ose që mund të keni bërë ndonjë gabim gjatë kopjimit të komandave, ose që nuk e keni vendosur ende gjuhën e saktë për komandat e TurtleScript (këtë mund ta bëni duke zgjedhur **Rregullimet** → **Gjuha e Programit**).

Ndoshta e keni kuptuar se `ec_përpara 100` e ka udhëzuar breshkën për të ecur përpara me 100 hapa (të vogla prej breshke), duke lënë edhe një vijë aty ku ka kaluar, dhe se `kthehu_majtas 90` e ka udhëzuar breshkën të kthehet me 90 gradë në të majtë.

Ju lutem shikoni adresat e mëposhtme në manualin e referimit për një shpjegim të plotë të komandave të reja: [ec_përpara](#), [ec_mbrapsht](#), [kthehu_majtas](#), dhe [kthehu_djathtas](#).

3.1.2 Shembuj të tjerë

Shembulli i parë ishte shumë i thjeshtë, kështu që le të shkojmë më tej!

```
rifillo

madhësi_tabele 200,200
ngjyrë_tabele 0,0,0
ngjyrë_lapsi 255,0,0
majë_lapsi 5

shko 20,20
drejtimi 135

ec_përpara 200
kthehu_majtas 135
ec_përpara 100
kthehu_majtas 135
ec_përpara 141
kthehu_majtas 135
ec_përpara 100
kthehu_majtas 45

shko 40,100
```


Udhëzuesi i KTurtle

Përsëri ju mund ti shkruani apo ti kopjoni komandat në shkrimore, ose të hapni shembullin shigjetë në menunë **Shembuj**, dhe ti zbatoni ato (duke përdorur **NiseNise**) për të parë rezultatin. Në shembujt e ardhshëm pritet që ti dini këto gjëra.

Mund të keni vënë re se ky shembulli i dytë përdor shumë më tepër rreshta. Keni parë gjithashtu edhe disa komanda të reja. Ja një shpjegim të shkurtër për të gjitha komandat e reja:

Pas komandës `rifillo` çdo gjë bëhet siç ishte kur sapo kishte filluar KTurtle.

`madhësi_tabele 200.200` vendos gjerësinë dhe lartësinë për tabelën e vizatimit në 200 pixel. Gjerësia dhe lartësia janë të barabarta, kështu që hapësira e vizatimit do të jetë një katror.

`ngjyrë_tabele 0,0,0` e bën hapësirën të zezë. `0,0,0` është një kombinim-**RGB** ku të gjitha vlerat janë vendosur në 0, gjë që rezulton si ngjyrë e zezë.

`ngjyrë_lapsi 255,0,0` përcakton ngjyrën e lapsit si të kuqe. `255,0,0` është një kombinim-**RGB** ku vetëm vlera e kuqe është e vendosur në 255 (plotësisht e ndezur), ndërsa të tjerat (e gjelbër dhe blu), janë të vendosur në 0 (plotësisht e fikur). Ky kombinim rezulton në një ngjyrë të kuqe të ndritshme.

Nëse nuk i kuptoni vlerat e ngjyrave, mos harroni të lexoni fjalorin për kombinimin-**RGB**.

`majë_lapsi 5` cakton gjerësinë (trashësinë) e majës së lapsit në 5 pixel. Tani e tutje çdo vijë që do të bëjë breshka do ta ketë trashësinë 5, deri sa `majë_lapsi` të ndryshohet në diçka tjetër.

`shko 20,20` urdhëron breshkën për të shkuar në një vend të caktuar në tabelë. E llogaritur që nga këndi i sipërm në të majtë, ky vend është 20 pixel përmas nga e majta, dhe 20 pixel poshtë kreut të tabelës. Vini re se kur përdoret komanda `shko` breshka nuk bën vijë.

`drejtimi 135` vendos drejtimin e breshkës. Komandat `kthehu_majtas` dhe `kthehu_djathtas` ndryshojnë këndin e breshkës, duke u nisur nga drejtimi i saj aktual. Komanda `drejtimi` e ndryshon këndin e breshkës duke filluar nga drejtimi zero, kështu që nuk ka lidhje me drejtimin e mëparshëm të breshkës.

Komanda `drejtimi` pasohet nga shumë komanda `ec_përpara` dhe `kthehu_majtas` të cilat bëjnë edhe vizatimin.

Më në fund një tjetër komandë `shko` është përdorur për ta çuar breshkën mënjane.

Sigurohuni që ti ndiqni lidhjet për te referenca. Referenca shpjegon çdo komandë më në tërësi.

Kapitulli 4

Referenca e Programimit për TurtleScript

Kjo është referenca për gjuhën TurtleScript të KTurtle. Në pjesën e parë të këtij kapitulli do të hedhim një vështrim në disa aspekte të [gramatikës](#) së gjuhës TurtleScript. Pjesa e dytë merret veçanërisht me [veprimet matematikore](#), [veprimet logjike \(e vërtetë / e rreme\)](#) dhe [veprimet krahasuese](#). Pjesa e tretë është në thelb një listë gjigante e të gjitha [komandave](#), me shpjegimet përkatëse për secilën prej tyre. Pjesa e katërt shpjegon se si u [jepën vlera ndryshoreve](#). Së fundi, në pjesën e pestë shpjegojmë se si organizohet zbatimi i komandave me [deklarimet për kontrollin e zbatimit](#), dhe në pjesën e gjashtë si mund të krijoni ju vetë komandat tuaja me anë të deklarimit mëso.

4.1 Gramatika e TurtleScript

Si çdo gjuhë, TurtleScript ka lloje të ndryshme fjalësh dhe simbolesh. Në shqip ne bëjmë dallim midis foljeve (si 'të ecësh' apo 'të këndosh') dhe emrave (si 'motër' apo 'shtëpi'); ato përdoren për qëllime të ndryshme. TurtleScript është një gjuhë programimi; ajo përdoret për të udhëzuar KTurtle çfarë të bëjë.

Në këtë seksion shpjegohen shkurtimisht disa prej llojeve të ndryshme të fjalëve dhe simboleve të TurtleScript-it. Do shpjegojmë [komentet](#), [komandat](#) dhe tre lloje të ndryshme vlerash: [numrat](#), [vargjet](#) dhe [vlerat logjike \(e vërtetë / e rreme\)](#).

4.1.1 Komentet

Një program përbëhet nga udhëzimet (apo komandat) që zbatohen kur programi nis, dhe të ashtuquajturat 'komente'. Komentet nuk zbatohen, KTurtle thjesht i injoron ato gjatë zbatimit të programit tuaj. Komentet janë për programuesit e tjerë, për t'i bërë ata të kuptojnë më mirë programin tuaj. Çdo gjë që pason një simbol # konsiderohet si koment në TurtleScript. Për shembull, ky program i vogël që nuk bën asgjë:

```
# Ky program i vogël është vetëm një koment dhe nuk bën asgjë!
```

Ai është i padobishëm por e shpjegon çështjen mirë.

Komentet bëhen shumë të dobishëm kur programi bëhet pak më i komplikuar. Ato mund të ndihmojnë për të dhënë disa këshilla për programuesit e tjerë. Në programin e mëposhtëm ju shihni komente që përdoren bashkë me komandën [shtyp](#).

Udhëzuesi i KTurtle

```
# Ky program është bërë nga Cies Breijs.  
shkruaj "ky tekst do të shkruhet në tabelë"  
# rreshti më sipër nuk është një koment, por rreshti pasues është:  
# shkruaj "ky tekst nuk do të shkruhet!"
```

Rreshti i parë përshkruan programin. Rreshti i dytë zbatohet nga KTurtle dhe shkruan në tabelën e vizatimit: ky tekst do të shkruhet në tabelë. Rreshti i tretë është një koment. Dhe rreshti i katërt është një koment i cili përmban një copë TurtleScript. Në qoftë se simboli # do të hiqet në rreshtin e katërt, komanda e shkrimit do të zbatohet nga KTurtle. Programuesit thonë: komanda e shkrimit në vijën e katërt është 'komentuar'.

Rreshtat e komentuar theksohen me ngjyrë gri të hapur në [shkrimore](#).

4.1.2 Komandat

Duke përdorur komandat ju i tregoni breshkës ose KTurtle të bëj diçka. Disa komanda nevojë të marrin parametra, disa nxjerrin rezultate.

```
# ec_përpara është një komandë që merr një parametër, në këtë rast numrin ←  
100:  
ec_përpara 100
```

Rreshti i parë është një [koment](#). Rreshti i dytë përmban komandën `ec_përpara` dhe [numrin](#) 100. Numri nuk është pjesë e komandës, por konsiderohet si një 'input' ose 'marrje' për komandën.

Për një pasqyrë të detajuar të të gjitha komandave që pranon KTurtle shkoni te [lista e komandave](#). Komandat e veta (ose të brendshme) janë të theksuara në ngjyrë blu të errët.

4.1.3 Numrat

Ka shumë mundësi që ju e dini mjaft mirë se ç'janë numrat. Mënyra si përdoren numrat në KTurtle nuk ndryshon shumë nga gjuha e folur, apo nga matematika.

Kemi numrat e ashtuquajtur natyrorë: 0, 1, 2, 3, 4, 5, etj. Numrat negativë: -1, -2, -3, etj. Dhe numrat me pikë dhjetore, për shembull: 0.1, 3.14, 33.3333, -5.05, -1.0.

Numrat mund të përdoren në [veprimet matematikore](#) dhe [veprimet krahasuese](#). Po ashtu ata mund të ruhen në [ndryshore](#). Numrat theksohen me të kuqe të errët.

4.1.4 Vargjet

Së pari një shembull:

```
shkruaj "Përshëndetje, unë jam një varg."
```

Në këtë shembull shkruaj është një urdhër (komandë), ku `''Përshëndetje, unë jam një varg.''` është një varg. Vargjet fillojnë dhe mbarojnë me shenjën `''`, nga këto shenja KTurtle e di se është një varg.

Vargjet mund të vihet në [ndryshore](#), ashtu si [numrat](#). Megjithatë, ndryshe nga numrat, vargjet nuk mund të përdoren në [veprimet matematikore](#) ose në [veprimet krahasuese](#). Vargjet theksohen me të kuqe.

4.1.5 Vlerat e vërtetësisë (e vërtetë / e rreme)

Ka vetëm dy vlera vërtetësie: e vërtetë dhe e rreme. Nganjëherë ato quhen edhe: ndezur dhe fikur, po dhe jo, një dhe zero. Por në TurtleScript ne i quajmë ato gjithmonë, e_vërtetë dhe e_rreme. Hidhni një sy në kësaj pjese TurtleScript:

```
$a = e_vërtetë
```

Nëse shikoni në [shqyrtues](#) ju mund të vini re se [ndryshore](#)\$a i është dhënë vlerae_vërtetë, dhe është e tipit vërtetësi.

Shpesh vlerat e vërtetësisë janë rezultat i një [veprimi krahasues](#), si në pjesën TurtleScript më poshtë:

```
$përgjigje = 10 > 3
```

[Ndryshorja](#)\$përgjigje merr vlerën e_vërtetë sepse numri 10 është më i madh se numri 3.

Vlerat e vërtetësisë, e_vërtetë dhe e_rreme theksohen me të kuqe të errët.

4.2 Veprimet matematikore dhe krahasuese

Titulli i kësaj pjese mund të tingëllojë pak si i vështirë, por në fakt nuk është aq e vështirë sa duket.

4.2.1 Veprimet matematike

Këto janë simbolet themelore matematike të njohur si: mbledhje (+), zbritje (-), shumëzim (*), pjesëtim (/) dhe ngritje në fuqi (^).

Ja një shembull i vogël i veprimeve matematikore që ju mund të përdorni në TurtleScript:

```
$mbledhje = 1 + 1
$zbritje = 20 - 5
$shumëzim = 15 * 2
$pjesëtim = 30 / 30
$fuqi = 2 ^ 2
```

Vlerat që përftohen nga këto veprime matematikore [u janë caktuar disa ndryshoreve](#). Duke përdorur [shqyrtuesin](#) ju mund të shihni vlerat e këtyre ndryshoreve.

Nëse doni të bëni vetëm një llogaritje të thjeshtë, mund të bëni diçka të tillë:

```
shkruaj 2010-12
```

Tani një shembull me kllapa:

```
shkruaj ( (20 - 5) * 2 / 30) + 1
```

Shprehjet brenda kllapave do të llogariten të parat. Në këtë shembull, do të llogaritet fillimisht 20-5, pastaj do të shumëzohet me 2, do të pjesëtohet me 30, dhe pastaj do t'i shtohet 1 (duke dhënë 2). Kllapat mund të përdoren edhe në raste të tjera.

KTurtle ka edhe veçori më të përparuara matematikore, në formën e komandave. Hidhuni një sy komandave të mëposhtme, por të jenë të vetëdijshëm se kanë lidhje me veprime të përparuara: [rumbullakos](#), [rastësisht](#), [sqrt](#), [pi](#), [sin](#), [cos](#), [tan](#), [arcsin](#), [arccos](#), [arctan](#).

4.2.2 Veprimet logjike të vërtetësisë (e vërtetë / e rreme)

Ndërsa [veprimet matematikore](#) janë kryesisht për [numrat](#), veprimet logjike janë për [vlerat e vërtetësisë](#) (e_vërtetë dhe e_rreme). Ka vetëm tre veprime logjike, të cilat janë: edhe, ose, dhe jo. Fragmenti TurtleScript i mëposhtëm tregon se si përdoren ato:

```
$edhe_1_1 = e_vërtetë edhe e_vërtetë # -> e_vërtetë
$edhe_1_0 = e_vërtetë edhe e_rreme   # -> e_rreme
$edhe_0_1 = e_rreme edhe e_vërtetë   # -> e_rreme
$edhe_0_0 = e_rreme edhe e_rreme     # -> e_rreme

$ose_1_1 = e_vërtetë ose e_vërtetë  # -> e_vërtetë
$ose_1_0 = e_vërtetë ose e_rreme     # -> e_vërtetë
$ose_0_1 = e_rreme ose e_vërtetë     # -> e_vërtetë
$ose_0_0 = e_rreme ose e_rreme       # -> e_rreme

$jo_1 = jo e_vërtetë   # -> e_rreme
$jo_0 = jo e_rreme     # -> e_vërtetë
```

Duke përdorur [shqyrtesin](#) ju mund të shihni vlerat e ndryshoreve, por rezultatet e veprimeve janë dhënë gjithashtu si komente të vogla në fund të rreshtave. Veprimi edhe jep vlerën e_vërtetë vetëm nëse të dyja anët kanë vlerën e_vërtetë. Veprimi ose jep vlerën e_vërtetë kur të paktën njëra anë është e_vërtetë. Kurse veprimi jo kthen vlerën e_vërtetë në e_rreme dhe vlerën e_rreme në e_vërtetë.

Veprimet logjike theksohen me rozë.

4.2.2.1 Disa shembuj më të përparuar

Shqyrtoni shembullin e mëposhtme me edhe:

```
$a = 1
$b = 5
nëse (($a < 10) edhe ($b == 5)) edhe ($a < $b) {
  shkruaj "tungjatjeta"
}
```

Në këto rreshta TurtleScript, përfundimet e tre [veprimeve krahasuese](#) janë bashkuar me anë të veprimit edhe. Kjo do të thotë që të treja duhet të japin vlerën "e_vërtetë", me qëllim që të shkruhet "tungjatjeta".

Një shembull me ose:

```
$n = 1
nëse ($n < 10) ose ($n == 2) {
  shkruaj "tungjatjeta"
}
```

Në këto rreshta TurtleScript, ana e majtë e veprimit ose merr vlerën 'e_vërtetë', ana e djathtë vlerën 'e_rreme'. Meqë njëra nga të dy anët e veprimit ose është 'e_vërtetë', vlera që jep veprimi ose është 'e_vërtetë'. Kjo do të thotë që "tungjatjeta" do të shkruhet.

Dhe në fund një shembull me veprimin jo, i cili vlerën 'e_vërtetë' e bën 'e_rreme', dhe vlerën 'e_rreme' e bën 'e_vërtetë'. Hidhini një sy:

```
$n = 1
nëse jo ($n == 3) {
  shkruaj "Shëndet!"
} përndryshe {
  shkruaj "Plaç! ;-)"
}
```

4.2.3 Veprimet krahasuese

Shikoni këtë krahasim të thjeshtë:

```
$përgjigje = 10 > 3
```

Këtu numri 10 krahasohet me numrin 3 me anë të veprimit krahasues 'më i madh se'. Rezultati i këtij krahasimi, *vlera logjike* e_vërtetë, ruhet në *ndryshoren* \$përgjigje.

Të gjithë *numrat* dhe *ndryshoret* (që përmbajnë numra) mund të krahasohen me njëri-tjetrin me anë të veprimeve krahasuese.

Këtu janë të gjitha veprimet krahasuese të mundshme: Mos harroni se \$A dhe \$B duhet të jenë

\$A == \$B	e barabartë	përgjigja është 'e_vërtetë' nëse \$A është e barabartë me \$B
\$A != \$B	e ndryshme	përgjigja është 'e_vërtetë' nëse \$A është e ndryshme nga \$B
\$A > \$B	më e madhe	përgjigja është 'e_vërtetë' nëse \$A është më e madhe se \$B
\$A < \$B	më e vogël	përgjigja është 'e_vërtetë' nëse \$A është më e vogël se \$B
\$A >= \$B	më e madhe baraz	përgjigja është 'e_vërtetë' nëse \$A është më e madhe ose e barabartë me \$B
\$A <= \$B	më e vogël baraz	përgjigja është 'e_vërtetë' nëse \$A është më e vogël ose e barabartë me \$B

Tabela 4.2: Llojet e pyetjeve

numra ose *ndryshore* që përmbajnë numra.

4.3 Komandat

Duke përdorur komandat ju i tregoni breshkës ose KTurtle të bëjë diçka. Disa komanda kanë nevojë për të marrë të dhëna, disa nxjerrin rezultate. Në këtë ndarje do të shpjegojmë të gjitha komandat e veta (të brendshme) të KTurtle. Nga ana tjetër, duke përdorur *mëso*, ju mund të krijoni komandat tuaj. Komandat e veta për të cilat do të flasim këtu theksohen me ngjyrë blu të errët.

4.3.1 Lëvizja e breshkës

Ka komanda të ndryshme për ta lëvizur breshkën mbi ekran.

ec_përpara (pp)

```
ec_përpara X
```

Udhëzuesi i KTurtle

`ec_përpara X` e lëviz breshkën përpara me një sasi prej `X` pikësh. Kur lapsi është i ulur, breshka lë një gjurmë (bën një vijë) gjatë lëvizjes. `ec_përpara` mund të shkurtohet si `pp`

ec_mbrapsht (mb)

```
ec_mbrapsht X
```

`ec_mbrapsht X` e lëviz breshkën mbrapa me një sasi prej `X` pikësh. Kur lapsi është i ulur, breshka lë një gjurmë (bën një vijë) gjatë lëvizjes. `ec_mbrapsht` mund të shkurtohet si `mb`.

kthehu_majtas (mj)

```
kthehu_majtas X
```

`kthehu_majtas` i thotë breshkës që të kthehet me një madhësi prej `X` gradësh në të majtë. `kthehu_majtas` mund të shkurtohet si `mj`.

kthehu_djathtas (dj)

```
kthehu_djathtas X
```

`kthehu_djathtas` i thotë breshkës që të kthehet me një madhësi prej `X` gradësh në të djathtë. `kthehu_djathtas` mund të shkurtohet si `dj`.

drejtimi (drj)

```
drejtimi X
```

`drejtimi` cakton drejtimin e breshkës me një madhësi prej `X` gradësh duke filluar nga drejtimi zero, dhe kështu që nuk ka lidhje me drejtimin e mëparshëm të breshkës. Drejtimi zero është drejtimi nga poshtë lartë. `drejtimi` mund të shkurtohet si `drj`.

shko_në_qendër

```
shko_në_qendër
```

`shko_në_qendër` e çon breshkën në qendër të tabelës së vizatimit.

shko

```
shko X, Y
```

`shko` i thotë breshkës të shkojë në një vend të caktuar në tabelë. Ky vend është `X` pika larg nga ana e majtë e tabelës, dhe `Y` pika larg nga ana e sipërme e tabelës.

shko_x

```
shko_x X
```

Duke përdorur komandën `shko_x` breshka do të vendoset `X` pika larg anës së majtë të tabelës, duke qëndruar në të njëjtën lartësi.

shko_y

```
shko_y Y
```

Duke përdorur komandën `shko_y` breshka do të vendoset `X` pika poshtë anës së sipërme të tabelës, duke qëndruar në të njëjtën largësi nga ana e majtë.

SHËNIM

Duke përdorur komandat `shko`, `shko_x`, `shko_y` dhe `shko_në_qendër` breshka nuk do të bëjë vijë, pa marrë parasysh nëse lapsi është i ngritur apo i ulur.

4.3.2 Ku është breshka?

Ka dy komanda që na japin pozicionin e breshkës në ekran.

merr_x

`merr_x` kthen numrin e pikave nga ana e majtë e tabelës deri në vendin e tanishëm të breshkës.

merr_y

`merr_y` kthen numrin e pikave nga ana e sipërme e tabelës deri në vendin e tanishëm të breshkës.

4.3.3 Breshka ka një laps

Breshka ka një laps që bën një vijë kur breshka lëviz. Ka disa komanda për të kontrolluar lapsin. Në këtë pjesë do të shpjegojmë këto komanda.

ngri_lapsin (ngri)

```
ngri_lapsin
```

`ngri_lapsin` e ngre lapsin nga tabela. Kur lapsi është 'i ngritur' nuk do të bëhet vijë kur breshka lëviz. Shiko dhe `ul_lapsin`. `ngri_lapsin` mund shkurtohet si `ngri`.

ul_lapsin (ul)

```
ul_lapsin
```

`ul_lapsin` ngjesh lapsin poshtë në tabelë. Me lapës të ulur poshtë në tabelë, kur breshka lëviz lë pas një vijë. Shiko edhe `ngri_lapsin`. `ul_lapsin` mund të shkurtohet në `ul`.

majë_lapsi (mjl)

```
majë_lapsi X
```

`majë_lapsi` cakton trashësinë e majës së lapsit (gjerësinë e vijës) në një madhësi prej `X` pikash. `majë_lapsi` mund të shkurtohet si `mjl`.

ngjyrë_lapsi (ngjl)

```
ngjyrë_lapsi R,G,B
```

`ngjyrë_lapsi` cakton ngjyrën e lapsit. `ngjyrë_lapsi` merr si parametra një kombinim RGB. `ngjyrë_lapsi` mund të shkurtohet në `ngjl`.

4.3.4 Komandat për të kontrolluar tabelën e vizatimit

Ka disa komanda për të kontrolluar tabelën e vizatimit.

madhësi_tabele (mt)

```
madhësi_tabele X,Y
```

Me komandën `madhësi_tabele` ju mund të caktoni madhësinë e tabelës së vizatimit. Ajo merr X dhe Y si parametra, ku X është gjerësia e re e tabelës në pika (pixel), dhe Y është lartësia e re e tabelës në pika (pixel). `madhësi_tabele` mund të shkurtrohet si `mt`.

ngjyrë_tabele (ngjt)

```
ngjyrë_tabele R,G,B
```

`ngjyrë_tabele` cakton ngjyrën e tabelës së vizatimit. `ngjyrë_tabele` merr një kombinim RGB si input. `ngjyrë_tabele` mund të shkurtrohet si `ngjt`.

4.3.5 Komandat për të pastruar

Ka dy komanda për të fshirë tabelën pasi të keni bërë ndonjë rrëmujë.

pastro (fshi)

```
pastro
```

Me `pastro` ju mund të fshini të gjitha vizatimet në tabelë. Të gjitha gjërat e tjera mbesin: vendi dhe drejtimi i breshkës, ngjyra e tabelës, dukshmëria e breshkës, dhe madhësia e tabelës.

rifillo

```
rifillo
```

`rifillo` pastron shumë më tepër se sa komanda `pastro`. Pas një komande `rifillo` çdo gjë bëhet siç ishte kur sapo u jap KTurtle. Breshka është e vendosur në mes të ekranit, ngjyra e tabelës është e bardhë, breshka bën një vijë të zezë në tabelë dhe madhësia e tabelës është 400 x 400 pika.

4.3.6 Breshka është një figurinë

Së pari një shpjegim të shkurtër se çfarë janë figurinat. Figurinat janë fotografi të vogla që mund të lëvizin nëpër ekran, si i shohim shpesh në lojërat me kompjuter. Edhe breshka jonë është gjithashtu një figurinë. Për më tepër informacion shiko fjalorin të figurinat.

Në vazhdim do të gjeni një pasqyrë të plotë mbi të gjitha komandat për të punuar me figurinat.

[Versioni i tanishëm i KTurtle nuk mbështet akoma përdorimin e figurinave të tjera përveç breshkës. Në versionet e ardhshme ju do të mund ta ndërroni breshkën me diçka tjetër që ju pëlqen.]

shfaq_figurinën (shf)

Udhëzuesi i KTurtle

```
shfaq_figurinën
```

shfaq_figurinën e bën të breshkën përsëri të dukshme, pasi të ketë qenë e fshehur. shfaq_figurinën mund të shkurtohet si shf.

fshih_figurinën (fsh)

```
fshih_figurinën
```

fshih_figurinën fsheh breshkën. Kjo mund të përdoret nëse breshka nuk përshtatet me vizatim tuaj. fshih_figurinën mund të shkurtohet si fsh.

4.3.7 A mund të shkruaj breshka?

Përgjigja është: 'po'. Breshka mund të shkruaj: ajo shkruan çdo gjë që ju ti thoni.

shkruaj

```
shkruaj X
```

Komanda shkruaj përdoret për ti thënë breshkës të shkruajë diçka në tabelë. shkruaj merr numra dhe vargje si input. Ju mund të shkruani numra dhe vargje të ndryshme duke përdorur shenjën '+'. Shikoni një shembull të vogël:

```
$viti = 2003
$autori = "Cies"
shkruaj $autori + " e filloi projektin KTurtle në vitin " + $viti + " ←
dhe akoma punon me qejf në të!"
```

madhësi_shkrimi

```
madhësi_shkrimi X
```

madhësi_shkrimi cakton madhësinë e shkronjave që përdoren nga shkruaj. madhësi_shkrimi merr një të dhënë që duhet të jetë një numër. Madhësia caktohet në pika (pixel).

4.3.8 Komandat matematikore

Në vijim janë komandat matematikore më të përparuara të KTurtle.

rumbullakos

```
rumbullakos(x)
```

rumbullakos numrin e dhënë te numri më i afërt i plotë.

```
shkruaj rumbullakos(10.8)
ec_përpara 20
shkruaj rumbullakos(10.3)
```

Me këtë program breshka do të shkruajë numrat 11 dhe 10.

rastësisht (kot)

```
rastësisht X,Y
```

rastësisht është një komandë që merr të dhëna dhe kthen një rezultat. Si të dhëna janë dy numra, i pari (X) cakton vlerën më të vogël (minimumin) e rezultatit, i dyti (Y) cakton vlerën më të madhe (maksimumin). Rezultati është një numër i zgjedhur rastësisht, i cili është më i madh ose i barabartë me minimumin dhe më i vogël ose i barabartë me maksimumin. Ja një shembull i vogël:

```
rifillo
përsërit 500 {
  $x = rastësisht 1,100
  ec_përpara $x
  kthehu_majtas 100 - $x
}
```

Duke përdorur komandën rastësisht ju mund ti shtoni pak rrëmujë (kaos) programit tuaj.

sqrt

```
sqrt X
```

Komanda sqrt përdoret për të gjetur rrënjën katrore të një numri X.

pi

```
pi
```

Kjo komandë kthen vlerën e konstantes Pi, 3.14159.

sin, cos, tan

```
sin X
cos X
tan X
```

Këto tre komanda përfaqësojnë funksionet e famshme trigonometrike sin, cos dhe tan. Argumenti X që u jepet këtyre komandave është një **numër**.

arcsin, arccos, arctan

```
arcsin X
arccos X
arctan X
```

Këto komanda janë funksionet e kundërta të **sin**, **cos** dhe **tan**. Argumenti X që u jepet këtyre komandave është një **numër**.

4.3.9 Bashkëbisedimi me programin

Bashkëbisedimi me programin kryhet përmes disa dritarezave të vogla që i nxjerr programi, të cilat ose japin një informacion, ose pyesin për diçka. KTurtle ka dy komanda për bashkëbisedimin, të cilat janë `mesazh` dhe `pyet`.

mesazh

```
mesazh X
```

Komanda `mesazh` merr një **varg** si plotësues. Ajo shfaq një dritare bashkëbisedimi që përmban **vargun** e dhënë.

```
mesazh "Cies e filloi programin KTurtle në vitin 2003 dhe akoma merret me qejf në të!" ←
```

pyet

```
pyet X
```

Komanda `pyet` merr një **varg** si plotësues. Ajo e nxjerr këtë varg në një dritare bashkëbisedimi (si komanda `mesazh`), bashkë me një vend për përgjigjen. Pasi përdoruesi të ketë futur një **numër** ose një **varg** në të, rezultati mund të ruhet në një **ndryshore** ose ti kalohet si plotësuese një **komande** tjetër. Për shembull:

```
$merr = pyet "Në cilin vit keni lindur?"
$nxirr = 2003 - $merr
mesazh "Në vitin 2003 keni qenë " + $nxirr + " vjeç."
```

Nëse përdoruesi e ndërpret dritaren e bashkëbisedimit, ose nuk shkruan në të asgjë fare, **ndryshoreja** është bosh.

4.4 Vlerëdhënia e ndryshoreve

Fillimisht do tu hedhim një sy ndryshoreve, dhe më pas do të shohim si u jepen vlera këtyre ndryshoreve.

Emrat e ndryshoreve fillojnë gjithmonë me një '\$'. Në **shkrimore** ato theksohen me ngjyrë vjollcë.

Ndryshoret mund të përmbajnë ndonjë **numër**, **varg shkronjash** ose **vlerë vërtetësie** (**e_vërtetë** / **e_rreme**). Përmbajtja e një ndryshoreje caktohet me anë të veprimit të vlerëdhënies, `=`. Ndryshoreja do të vazhdojë ta ketë këtë vlerë deri në përfundim të programit, ose deri sa ti jepet një vlerë tjetër.

Pasi u është dhënë një vlerë, ndryshoret mund të përdoren njësoj si vetë vlera që ato përmbajnë. Shikoni për shembull rreshtat e mëposhtëm TurtleScript:

```
$x = 10
$x = $x / 3
shkruaj $x
```

Fillimisht ndryshore `$x` i jepet vlera 10. Pastaj `$x` i ri-jepet vlera e vetvetes e pjesëtuar për 3 — që do të thotë se vlera e re e `$x` është $10 / 3$. Më në fund, ndryshoreja `$x` shkruhet. Vini re që në rreshtin e dytë dhe të tretë `$x` është përdorur si të ishte vetë vlera që ajo përmban.

Që të përdoren ndryshoret duhet që fillimisht tu jepet një vlerë. Për shembull:

```
shkruaj $n
```

do të japë një njoftim gabimi, sepse ndryshorja \$n nuk ka asnjë vlerë.

Ju lutem shikoni me vëmendje rreshtat e mëposhtëm TurtleScript:

```
$a = 2004
$b = 25

# komanda pasuese shkruan "2029"
shkruaj $a + $b
ec_mbrapsht 30
# komanda pasuese shkruan "2004 plus 25 baraz 2029"
shkruaj $a + " plus " + $b + " baraz " + ($a + $b)
```

Në dy rreshtat e parë ndryshoret \$a dhe \$b kanë marrë vlerat 2004 dhe 25. Pastaj zbatohen dy komandat shkruaj me një komandë ec_mbrapsht 30 në mes. Komentet përpara komandave shkruaj shpjegojnë çfarë bëjnë këto komanda. Komanda ec_mbrapsht 30 është përdorur për tu siguruar që gjërat e shkruara janë në rreshta më vete. Siç e shihni, ndryshoret mund të përdoren si vetë përmbajtja që ato kanë, mund të përdoren me çdo lloj [veprimi](#) ose të jepen si plotësuese të [komandave](#) që thirren.

Edhe një shembull tjetër:

```
$emër = pyet "Si e ke emrin?"
print "Përshëndetje " + $emër + "! Suksese gjatë mësimit të artit të ←
      programimit..."
```

Mjaft i thjeshtë. Përsëri mund të shihni që ndryshorja \$emër, është përdorur si të ishte një varg shkronjash.

Kur përdoren ndryshoret, [shqyrtuesi](#) është shumë i dobishëm. Ai tregon përmbajtjen e të gjitha ndryshoreve që janë aktualisht në përdorim.

4.5 Kontrollimi i zbatimit

Kontrolluesit e zbatimit mundësojnë — siç e nënkupton edhe emri i tyre — që të kontrollohet rrjedha e zbatimit të komandave.

Komandat e kontrollit të zbatimit theksohen me të gjelbër të errët dhe me të trasha. Kllapat përdoren kryesisht së bashku me kontrolluesit e zbatimit dhe ato theksohen me të zezë.

4.5.1 Bëjeni breshkën të presë

Nëse keni programuar ndopak në KTurtle ju mund të keni vënë re se breshka mund të jetë shumë e shpejtë në vizatim. Kjo komandë e bën breshkën të presë për një sasi të caktuar kohe.

prit

```
prit X
```

prit e bën breshkën të presë për X sekonda.

```
përsërit 36 {
  ec_përpara 5
  kthehu_djathtas 10
  prit 0.5
}
```

Ky program vizaton një rreth, por breshka do të presë gjysmë sekonde pas çdo hap. Kjo të jep përshtypjen e një breshke që lëviz ngadalë.

4.5.2 Zbato “nëse”

nëse

```
nëse vërtetësi { ... }
```

Komandat që janë vendosur midis kllapave do të zbatohen vetëm nëse [vlera e vërtetësisë](#) është ‘e_vërtetë’.

```
$x = 6
nëse $x > 5 {
  shkruaj "numri $x është më i madh se pesa!"
}
```

Në rreshtin e parë \$x merr vlerën 6. Në rreshtin e dytë një [veprim krahasimi](#) është përdorur për të vlerësuar \$x > 5. Meqë ky jep vlerën ‘e_vërtetë’, 6 është më e madhe se 5, kontrolluesi i zbatimit nëse do të lejojë komandat midis kllapave që të zbatohen.

4.5.3 Nëse jo, ose me fjalë të tjera: “përndryshe”

përndryshe

```
nëse vërtetësi { ... } përndryshe { ... }
```

përndryshe mund të përdoret si plotësues i kontrolluesit të zbatimit [nëse](#). Komandat rrethuar nga kllapat, pas përndryshe, zbatohen vetëm nëse [vërtetësia](#) del ‘e_rreme’.

```
rifillo
$x = 4
nëse $x > 5 {
  shkruaj "numri $x është më i madh se pesë!"
} përndryshe {
  shkruaj "numri $x është më i vogël se gjashtë!"
}
```

[Veprimi i krahasimit](#) vlerëson shprehjen \$x > 5. Me që 4 nuk është më i madh se 5, shprehja merr vlerën ‘e_rreme’. Kjo do të thotë se do të zbatohen komandat mes kllapave pas përndryshe.

4.5.4 Cikli “përderisa”

përderisa

```
përderisa vërtetësi { ... }
```

Kontrolluesi i zbatimit përderisa është mjaft i ngjashëm me [nëse](#). Ndryshimi është se përderisa vazhdon të përsërit (në mënyrë ciklike) komandat midis kllapave, deri sa [vërtetësia](#) të bëhet ‘e_rreme’.

```

$x = 1
përderisa $x < 5 {
  ec_përpara 10
  prit 1
  $x = $x + 1
}

```

Në rreshtin e parë \$x merr vlerën 1. Në rreshtin e dytë vlerësohet \$x < 5. Meqenëse përgjigja e kësaj pyetje është ‘e_vërtetë’, kontrolluesi i zbatimit përderisa fillon zbatimin e komandave midis kllapave. Kjo punë përsëritet disa herë, derisa \$x < 5 bëhet ‘e_rreme’. Në këtë shembull, komandat midis kllapave do të zbatohen 4 herë, sepse çdo herë që zbatohet rreshti i pestë, \$x rritet me 1.

4.5.5 Cikli “përsërit”

përsërit

```
përsërit numër { ... }
```

Kontrolluesi i zbatimit përsërit është shumë i ngjashëm me [përderisa](#). Ndryshimi është se përsërit vazhdon të përsërisë (në mënyrë ciklike) komandat midis kllapave për aq herë sa numri i dhënë.

4.5.6 Cikli “për_çdo”, një cikël numëruar

për_çdo

```
për_çdo ndryshore = numër deri_në numër { ... }
```

Cikli për_çdo është një ‘cikël numëruar’, që do të thotë se numëron për ju. Numri i parë i cakton ndryshores vlerën e parë në cikël. Në çdo rrotullim numri rritet deri sa të arrijë numrin e dytë.

```

për_çdo $x = 1 deri_në 10 {
  shkruaj $x * 7
  ec_përpara 15
}

```

Sa herë që komandat midis kllapave zbatohen, ndryshorja \$x rritet me 1, deri sa \$x arrijë vlerën 10. Komandat midis kllapave shkruajnë \$x shumëzuar me 7. Pasi programi përfundon zbatimin, ju do të shihni në ekran tabelën e shumëzimit me 7.

Madhësia e hapit zakonisht është 1, por you mund të zgjidhni një vlerë tjetër.

```
për_çdo ndryshore = numër deri_në numër me_hap numër { ... }
```

4.5.7 Braktisja e një cikli

braktis

```
braktis
```

E përfundon menjëherë ciklin aktual dhe e kalon kontrollin te komanda që vjen menjëherë pas këtij cikli.

4.5.8 Ndaloni zbatimin e programit

dil

```
dil
```

Përfundon zbatimin e programit tuaj.

4.6 Krijoni komandat tuaja me 'mëso'

mëso është e veçantë sepse përdoret për të krijuar komandat tuaja. Komandat që krijoni mund të marrin të dhëna plotësuese dhe të kthejnë rezultate. Le të hedhim një sy se si krijohet një komandë e re:

```
mëso rreth $x {
  përsërit 36 {
    ec_përpara $x
    kthehu_majtas 10
  }
}
```

Komanda e re është quajtur `rreth`. Komanda `rreth` merr një të dhëna plotësuese, për të vendosur madhësinë e rrethit. `rreth` nuk kthen rezultat. Komanda `rreth` tani mund të përdoret si një komandë e zakonshme në pjesën tjetër të programit. Shikoni këtë shembull:

```
mëso rreth $x {
  përsërit 36 {
    ec_përpara $x
    kthehu_majtas 10
  }
}

shko 200,200
rreth 20

shko 300,200
rreth 40
```

Në shembullin tjetër krijohet një komandë e cila kthen një vlerë.

```
mëso faktorial $x {
  $r = 1
  për_çdo $i = 1 deri_në $x {
    $r = $r * $i
  }
  kthim $r
}

shkruaj faktorial 5
```

Në këtë shembull është krijuar një komandë e re e quajtur `faktorial`. Nëse kësaj komande i jepet 5 në hyrje, ajo na kthen në dalje $5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$. Duke përdorur `kthim`, përcaktohet vlera e daljes dhe rrjedha e zbatimit kthehet aty ku u thirr komanda.

Komandat mund të marrin më shumë se një vlerë hyrëse. Në këtë shembull krijohet një komandë që vizaton një drejtkëndësh:

Udhëzuesi i KTurtle

```
mëso kuti $x, $y {  
  ec_përpara $y  
  kthehu_djathtas 90  
  ec_përpara $x  
  kthehu_djathtas 90  
  ec_përpara $y  
  kthehu_djathtas 90  
  ec_përpara $x  
  kthehu_djathtas 90  
}
```

Tani ju mund të zbatoni kuti 50, 100 dhe breshka do të vizatojë një drejtkëndësh në tabelë.

Kapitulli 5

Fjalor

Në këtë kapitull do të gjeni një shpjegim për shumicën e fjalëve ‘të pazakonta’ që janë përdorur në udhëzues.

gradët Gradët janë njësi për matjen e këndeve ose rrotullimeve. Një rrotullim i plotë është 360 gradë, një gjysmë rrotullim 180 gradë dhe një çerek rrotullim 90 gradë. Komandat `kthehu_majtas`, `kthehu_djathtas` dhe `drejtimi` kanë nevojë për një të dhënë hyrëse në gradë.

të dhënat hyrëse dhe dalëse të komandave Disa komanda marrin të dhëna, disa komanda nxjerrin të dhëna, disa komanda edhe marrin edhe nxjerrin të dhëna, dhe disa komanda as nuk marrin dhe as nuk nxjerrin të dhëna.

Disa shembuj të komandave që vetëm marrin të dhëna janë:

```
ec_përpara 50
ngjyrë_lapsi 255,0,0
shkruaj "Ç'kemi!"
```

Komanda `ec_përpara` merr 50 si të dhënë hyrëse. `ec_përpara` ka nevojë për këtë të dhënë plotësuese që të dijë se sa pika (pixel) duhet të shkojë përpara. `ngjyrë_lapsi` merr një ngjyrë si të dhënë hyrëse dhe `shkruaj` merr një varg (një pjesë teksti) si të dhënë hyrëse. E dhëna hyrëse mund të jetë edhe një enë. Shembull tjetër e ilustron këtë:

```
$x = 50
shkruaj $x
ec_përpara 50
$str = "Ç'kemi!"
shkruaj $str
```

Tani disa shembuj komandash që japin të dhëna:

```
$x = pyet "Të lutem shkruaj diçka dhe shtyp OK... faleminderit!"
$r = rastësisht 1,100
```

Komanda `pyet` merr një varg si të dhënë hyrësedhe nxjerr numrin apo vargun që është shkruar si përgjigje. Siç mund ta shikoni, rezultati që jep komanda `pyet` është ruajtur në enën `x`. Komanda `rastësisht` gjithashtu jep një të dhënë. Në këtë rast ajo nxjerr një numër nga 1 deri në 100. Rezultati i komandës `rastësisht` është ruajtur përsëri në një enë, të quajtur `r`. Vini re se enët (mbajtëset e të dhënave) `x` dhe `r` nuk janë përdorur në shembullin e mësipërm.

Ka edhe komandave që as nuk marrin dhe as nuk japin të dhëna. Ja disa shembuj:

Udhëzuesi i Kturtle

```
pastro  
ngri_lapsin
```

theksimi ndihmues Ky është një tipar i Kturtle që e bën edhe më të lehtë programimin. Me theksimin ndihmues kodi që ju shkruani merr një ngjyrë që tregon se çfarë lloj kodi është. Në listën e mëposhtme do të gjeni llojet e ndryshme të kodit dhe ngjyrat që marrin në [shkrimore](#).

pika (pixels) Një pixel është një pikë në ekran. Nëse do ta shikoni nga shumë afër, do të vini re se ekranin e monitorit tuaj përdor pika. Të gjitha figurat në ekran janë ndërtuar me anë të këtyre pikave. Një pixel është gjëja më e vogël që mund të vizatohet në ekran.

Ka shumë komanda që u duhet një numër pikash si e dhënë hyrëse. Këto komanda janë: `ec_përpara`, `ec_mbrapsht`, `shko`, `shko_x`, `shko_y`, `madhësi_tabele` dhe `majë_lapsi`.

Në versionet e hershme të Kturtle tabela e vizatimit ishte në thelb një imazh përfundimtar, por në versionet e fundit tabela është një vizatim vektorial. Kjo do të thotë që hapësira e vizatimit mund të zmadhohet dhe të zvogëlohet, prandaj një pixel nuk është e thënë që të jetë patjetër një pikë në ekran.

kombinimet RGB (kodet e ngjyrave) kombinimet RGB përdoren për të përshkruar ngjyrat. 'R' përfaqëson 'të kuqen (Red)', 'G' përfaqëson 'të gjelbrën (Green)' dhe 'B' përfaqëson 'blunë (Blue)'. Për shembull ja një kombinim RGB: 255, 0, 0. Vlera e parë ('e kuqe') është 255 dhe të tjerat janë 0, kështu që kjo paraqet një nuancë të kuqe të fortë. Çdo vlerë në një kombinim RGB duhet të jetë nga 0 deri në 255. Ja një listë e vogël e disa ngjyrave që përdoren shpesh:

Dy komanda kërkojnë një kombinim RGB si të dhënë hyrëse; këto komanda janë `ngjyrë_tabele` dhe `ngjyrë_lapsi`.

figurina Një figurinë është një foto e vogël që mund të lëvizet nëpër ekran. Breshka jonë e dashur, për shembull, është një figurinë.

Shënim: në këtë version të Kturtle figurina nuk mund të ndryshohet nga një breshkë në diçka tjetër. Në versionet e ardhshme të Kturtle do të jetë e mundur të bëhet kjo.

Udhëzuesi i KTurtle

komandat e rregullta	blu e errët	Komandat e rregullta janë përshkruar këtu .
komandat e kontrollit të zbatimit	e zezë (e trashë)	Këto komanda të veçanta kontrollojnë radhën e zbatimit të komandave të tjera, lexoni më shumë mbi to këtu .
komentet	gri	Rreshtat që janë komentuar fillojnë me një shenjë komenti (#). Këto rreshta nuk përfillen gjatë zbatimit të programit. Komentet i japin mundësi programuesit që të shpjegojë pak në lidhje me kodin e tij. Por mund të përdoren edhe për të ç'aktivizuar përkohësisht një pjesë të caktuar të kodit (që të mos zbatohet).
klapat {,}	e gjelbër e errët (me të trashë)	Kllapat përdoren për të grupuar pjesë të programit. Kllapat përdoren shpesh bashkë me kontrolluesit e zbatimit .
komanda mëso	e gjelbër e çelur (me të trashë)	Komanda mëso përdoret për të krijuar komanda të reja.
vargjet	e kuqe	Nuk ka shumë për të thënë edhe për vargjet (shkronjore), përveç se ata gjithmonë fillojnë dhe mbarojnë me thonjëzat dyshe ("").
numrat	e kuqe e errët	Nuk kemi ç'të themi ndonjë gjë për numrat.
vlerat e vërtetësisë	e kuqe e errët	Ka vetëm dy vlera vërtetësie, që janë: e_vërtetë dhe e_rreme.
ndryshoret	vjollcë	Ndryshoret fillojnë me shenjën e dollarit '\$'. Ato janë si enë që mund të përmbajnë vlera numerike (numra), vargje apo vlera vërtetësie.
veprimet matematikore	gri	Veprimet matematikore janë këto: +, -, *, / dhe ^.
veprimet krahasuese	bojë-qielli (me të trashë)	Veprimet krahasuese janë këto: ==, !=, <, >, <= dhe >=.
veprimet e vërtetësisë	rozë (me të trashë)	Veprimet e vërtetësisë janë këto: edhe, ose dhe jo.
teksti i zakonshëm	i zi	

Tabela 5.2: Llojet e ndryshme të kodit dhe ngjyrat e theksimit të tyre

Udhëzuesi i Kturtle

0,0,0	i zi
255,255,255	e bardhë
255,0,0	e kuqe
150,0,0	e kuqe e errët
0,255,0	e gjelbër
0,0,255	blu
0,255,255	blu e çelur
255,0,255	rozë
255,255,0	e verdhë

Tabela 5.4: Kombinime RGB që përdoren shpesh

Kapitulli 6

Udhëzuesi i Përkthyesit të KTurtle

Siç mund ta dini tashmë, gjuha e programimit të KTurtle TurtleScript, lejon të jenë e përkthyer. Kjo largon një pengesë veçanërisht për nxënësit e vegjël, në përpjekjet e tyre për të kuptuar bazat e programimit.

Gjatë përkthimit të KTurtle në një gjuhë tjetër, përveç vargjeve të ndërfaqes GUI, ju do të gjeni edhe komandat e programimit, shembujt dhe mesazhet e gabimit, të cilët janë përfshirë në skedarin standard .pot që përdoret për përkthim në KDE. Çdo gjë përkthehet duke përdorur metodën e zakonshme të përkthimit në KDE, e megjithatë është shumë e këshillueshme që të mësohet pak se si duhen përkthyer këto (siç do të lexoni gjithashtu edhe në komentet për përkthyesin).

Ju lutem shikoni në <http://edu.kde.org/kturtle/translator.php> për më shumë informacion në lidhje me procesin e përkthimit. Shumë faleminderit për punën tuaj! KTurtle varet shumë nga përkthimet e tij.

Kapitulli 7

Vlerësimet dhe Liçenca

KTurtle

Software copyright 2003-2007 Cies Breijs cies AT kde DOT nl

Documentation copyright 2004, 2007, 2009

- Cies Breijs cies AT kde DOT nl
- Anne-Marie Mahfouf annma AT kde DOT org
- Disa redakte nga Philip Rodrigues phil@kde.org
- Aktualizim i udhëzuesit të përkthimit dhe disa redakte nga Andrew Coles andrew_coles AT yahoo DOT co DOT uk

Përkthyer në shqip për herë të parë nga Dashamir Hoxha dashohoxha AT gmail DOT com

Ky dokument shpërndahet sipas kushteve të Liçencës GNU për Dokumentacionin e Lirë.

Ky program shpërndahet sipas kushteve të liçencës GNU General Public License.

Shtesa A

Instalimi

A.1 Ku ta gjejmë KTurtle

KTurtle është pjesë e projektit KDE <http://www.kde.org/> .

KTurtle gjendet në paketën kdeedu në <ftp://ftp.kde.org/pub/kde/> , burimi kryesor i FTP për projektin KDE.

A.2 Përpunimi dhe Instalimi

Për sqarime të hollësishme se si kompilohen dhe instalohen programet KDE shiko [Building KDE4 From Source](#)

Meqënëse KDE përdor **cmake** normalisht nuk duhet të jetë e vështirë për ta kompiluar. Nëse ju dalin probleme, i raportoni te listat e diskutimeve të KDE.