Opis projekta

Bubble Trouble

Dominik Mikulčić, Lorena Novak, Petra Škrabo, Tihana Štifanić, Katarina Šupe

Retro igra izrađena u Processingu napravljena po uzoru na igru hrvatskog kreatora Krešimira Cvitanovića. Igrač je predstavljen likom koji puca u mjehuriće koji odskaču od tlo i postaju manji ukoliko ih igrač pogodi. Svaki put kada igrač pogodi mjehurić, on se razdvoji u nekoliko manjih, a na igraču je da pripazi da ga ti manji mjehurići također ne pogode. Svaki level ima vremenski limit u kojem igrač mora pokušati uništiti sve mjehuriće što brže.

Predviđeni datum završetka rada: 1.4.2021.

Poveznica na projekt na Githubu: <https://github.com/katarinasupe/Bubble-Trouble>

Dnevnik rada

* 7. ožujka 2021. (Tihana Štifanić): Stvaranje klase *Player* koja predstavlja igrača. Igrač je određen svojom pozicijom *position* (po *x*-koordinati, kako je pomicanje moguće samo u tom smjeru) i brojem preostalih života *lives*. U klasi se također pamte podaci o igračevom koplju (trenutna pozicija glave koplja određena s *xSpear* i *ySpear* te neke pomoćne varijable koje označavaju je li koplje trenutno aktivno i u kojem smjeru se trenutno pomiče). Koplje igrača se aktivira spacebarom, a igrač njime može „razbijati“ loptice koje su trenutno na ekranu. Također su implementirane metode za iscrtavanje igrača i koplja *draw()*, metoda za pomicanje igrača za *x* piksela *move()* te metoda za ažuriranje *update()* u kojoj se trenutno samo ažurira pozicija glave koplja (kada je koplje aktivno, glava se prvo pomiče prema gore dok ne udari u gornji rub prozora, a tada prema dolje dok ne udari u donji rub, kad postaje neaktivno). Implementirane su i metode za resetiranje pozicije igrača *resetPosition()* te resetiranje koplja *resetSpear()*. Trenutno su sve metode implementirane sa samo jednim igračem u vidu, pa ih treba prilagoditi za igru s dva igrača. – 3 sata
* 8. ožujka 2021. (Tihana Štifanić):
  + Stvaranje klase *Ball* koja predstavlja lopticu. Loptica je određena svojim središtem (*xCenter*, *yCenter*) i radijusom *radius*. U igri postoji samo nekoliko mogućih veličina loptice pa je uveden atribut *sizeLevel*, o kojem ovisi radijus loptice. Najmanje loptice koje se pojavljuju u igri su one kod kojih je *sizeLevel = 1*, a predviđeno je da su najveće loptice koje se pojavljuju one kod kojih je *sizeLevel = 6*. Kad god igrač pogodi neku lopticu svojim kopljem, ona se razdvaja u dvije loptice čiji je *sizeLevel* za jedan manji. Kada igrač pogodi lopticu za koju je *sizeLevel = 1*, ona nestaje. Trenutno su *radius* i *sizeLevel* povezani tako da je *radius = sizeLevel \* 10*. Loptice također imaju brzinu i smjer kretanja određen s (*xVelocity, yVelocity*) koji se zbraja sa središtem loptice prilikom svakog pomaka u metodi *update()*. U metodi *update()* se također provjerava je li loptica udarila u neki od rubova ekrana, čime se njena brzina mijenja na odgovarajući način. Osim toga, implementirane su metode *draw()* za iscrtavanje loptice te *checkSpearCollision()* i *checkPlayerCollision()* koje primaju koordinate glave koplja, odnosno poziciju igrača te provjeravaju nalaze li se one u radijusu oko središta loptice. – 5 sati
  + U glavnom programu su napisane metode koje se brinu za pomicanje igrača, *keyPressed()* i *keyReleased()* te dvije pomoćne preopterećene metode *setMove()*. Također je dodana varijabla *state* koja se koristi za postavljanje i određivanje trenutnog stanja programa, a planirano je da bude u jednom od stanja *MAINMENU*, *SETTINGS* ili *GAME*, definiranih u enumeraciji *State*. U glavnoj *draw()* funkciji se potom iscrtavaju samo oni elementi ili pozivaju one metode koje pripadaju jednom od stanja (za sad su to samo elementi stanja *GAME*). Tako se u glavnoj *draw()* funkciji prilikom stanja *GAME* iscrtavaju loptice, igrač te se provjerava postoje li kolizije. U svrhu provjera kolizija, implementirane su metode *ballSpearCollision()* te *ballPlayerCollision()* koje prvo za sve postojeće loptice pozivaju ranije spomenute metode *Ball.checkSpearCollision()* i *Ball.checkPlayerCollision()* te obrađuju posljedice kolizije (nestajanje ili razdvajanje loptice, odnosno gubitak života). Obrada posljedica (npr. završetak igre – *Game over*) nije još u potpunosti implementirana. – 4 sata
* 10. ožujka 2021. (Dominik Mikulčić): implementirano dijeljenje lopte kad ona udari koplje bilo gdje ispod vrha (riječ je o malim promjenama u funkciji *Ball.CheckSpearCollision*), potom je prilagođeno oduzimanje života (dodavanjem atrubuta *is\_being\_hit* klasi *Ball* koji pamti udara li lopta igrača) te su zatim funkcionalnosti igre prilagođene za dva igrača (dodavanjem liste *players* koja sadrži sve igrače pa se funkcionalnosti implementiraju na sličan način kao kod liste *balls* i dodavanjem pomoćne varijable *quantity* koja pamti broj igrača) – 6 sati