Forritun

Kafli 1, tími 1:

* Python er túlkað mál
* Ef þú setur input(“blabla”) þá færðu alltaf út streng
* Int er heiltala
* Float er kommutala
* Str er strengur
* Notum oft import math (math.pi og svo frv.)
* = þýðir ekki jafnaðarmerki, ‘gildi’ vinstra megin við = og svo ‘breyta’ hægra megin
* Notum \ til að halda áfram með setningu sem fer í línuna fyrir neðan
* Nöfn sem við búum til (breytur) mega ekki byrja á tölustöfum og nöfn sem byrja á \_ hafa sérstaka þýðingu
* / = deiling
* // = deiling og fáum út heila tölu
* % = hversu of gengur talan upp í hina töluna 🡪 þetta þýðir afgangur af því

Kafli 2, tími 2:

* “Selections is how program makes choices”
* = er ‘jafnt og’ merki og != er ‘ekki jafnt og’ (boolean)
* boolean = true/false og if og else setningar og fl.
* if true þá keyra allar setningar
* ‘Element of the suit’ þurfa alltaf að vera inndregnar og allar inndregnar jafn mikið (oftast 4 bil) svo að python þekki þetta (lesi forritið rétt)
* Þegar við erum að vinna mikið með endurtekningar í forritun þá notum við frekar while og for lykkjuna
* Þannig while virkar alltaf á meðan ‘ástandið, setningin’ er true og hún getur haldið endalaust áfram ef setningin er alltaf sönn
* Outside the loop, initialize the boolean og inn í lykkjunni (while-lykkja) þá býrðu til einhverja aðgerð sem að endanum leiðir til “false” og þetta tvennt þarf að vera
* Fullkomnar tölur er það að summa talna er jafnt og talan, fyrsta fullkomna talan er 6(1+2+3)
* Abundant numbers summed to more than the number, -12:1+2+3+4+6=16 ???
* Deficient numbers summed to less than the number, -13:1 ???
* Boolean true og false skilyrði- dæmi um true: 1, 100, “hello”, [a,b] og dæmi um false: 0, “ “, [ ]
* None er lika gildi sem er minna en allt og jafnt og ekkert
* Við notum id() fyrir Equality
* Abs er notað fyrir “close enough”
* And, or og not eru þær þrjár aðgerðir sem við notum með boolean
* Reationals skila alltaf true eða false
* Compound statements = fjölsetningar eru margar setningar notaðar í “grúbbu”
* Boolean segðin hættir í lykkjunni um leið og hún fær út true og sleppir þá hinum “if”
* Það er hægt að nota while lykkju með else
* Það er hægt að nota “break statement” í lykkju og þá er farið strax úr henni
* The range function represents a sequence of integers and it takes 3 arguments

Kafli 2, tími 3 (sami glærupakki)

* A continue statement í while lykkju: fara strax upp í byrjun lykkjunar
* A break statement í while lykkju: fara strax úr lykkjunni og skippa rest ef það er eitthvað eftir
* The range argument takes 3 arguments: 1- byrjun á ranginu, 0 ef ekkert annað sé gefið upp, 2-endirinn á rangin, upp að tölunni en ekki jafnt og hún, 3-búist við sé 1 ef ekkert er gefið upp
* Ef það er bara eitt argument gefið þá er búist við því að það sé enda argumentið (the end value)

Kafli 2, tími 4 (sami glærupakki)

* “The keyword” fyrir for lykkjur er ‘In’
* Í for lykkjum er líka notast við else, break og continue
* Við notum range function i for lykkum, td (1,100) en það getur tekið alveg upp í 3 gildi, þetta range function er hálfopið bil þannig að talan lengst til hægri er aldrei notuð með í dæminu
* Ef það er bara ein tala í range function þá er 0 alltaf byrjunar talan sem vinnur sig síðan upp í töluna sem er í range-inu
* Þegar það eru 3 gildi i range-inu þá eru fyrstu 2 tölurnar bilið sem tölurnar eru á en sú þriðja er hvað er mikið bil á milli talnanna

Kafli 3, tími 5

* An algorithm is a description of how to solve a problem
* A program is an implementation (útfærsla) of an algorithm in a particular language to run on computer (usually a particular kind of computer
* Git init = initialixe local git repository
* Git add <file> = add files to index
* Git status = check status of working tree
* Git commit = commit changes in index
* Git push = push to remote repository
* Git pull = pull latest from remote repository
* Git clone = clone repository into a new directory
* Git config –global user.name = ‘Thora’, notað til að skýra
* Git rm –cached nafn á file = telur file-ið úr möppunni
* Git add . = setja allt í möppuna (. = all)