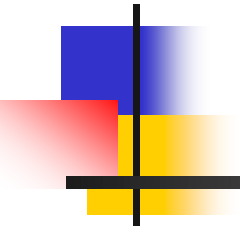


Python

Informatica

Renske Weeda

Plan van vandaag

- Ontwerp maken voor een complexe programma (een spel)
- Grote taak opdelen in kleinere taken
- Uitleg werkomgeving (Pycharm?)
- Uitleg PO

VWO4 PO Galgje

- Opdracht omschrijving komt op site.

- Gebruik hiervoor je kennis over:
  - Lijsten
  - While-loops

Complexe programma: aanpak

- Niet meteen programmeren:
  - Veel fouten → tijd en frustratie!
  - Code wordt onlogisch: onleesbaar spaghetti, moeilijk uit te breiden of aan te passen
  - Code niet efficiënt: veel code dubbel
  - Probleem lijkt moeilijk: eerst opdelen in kleinere stukken die je los van elkaar kunt programmeren en testen
- Eerst een ontwerp!
  - Stroomdiagram (op hoog niveau – dus eerst zonder code details)
  - Opsomming van data (wat voor dingen moet je opslaan)

PO ontwerp

- Om te oefenen maken we nu eerst samen het ontwerp voor het spel: gok de getallen

Spel: Gok de getallen

- Stel we spelen een spel
- Computer heeft een lijstje getallen in ‘gedachten’

- Doel: Speler moet alle getallen raden
  - Speler wint als **alle** getallen worden geraden
  - Speler verliest zodra één verkeerd getal geraden wordt

Spel: Gok de getallen

Het spelverloop:

- Maak een lijst met een paar getallen er in
- Zolang het spel niet is afgelopen:
  - Vraag de gebruiker om een getal in te typen
  - Als het getal voorkomt in de lijst, haal je het getal eruit
  - Als het getal niet voorkomt in de lijst, is het spel afgelopen.

Tips:

- Bepaal welke variabelen nodig om te onthouden
- Teken een stroomdiagram
- In je spel zit herhaling, maak daarom gebruik van een while-loop
- Gebruik een Boolean vlag, spel_is_afgelopen. Voordat je begint met spelen staat deze op False. Je spel herhaalt zich zolang deze op False staat. Zet het op het juiste moment op True.

Spel: Gok De Getallen

- Wat moeten we steeds doen?
  - Controleren of spel is afgelopen
  - Gebruiker om een gok vragen
  - Controleren of gok in lijst voorkomt
- We gaan steeds door zolang:
  - NIET spel_is_afgelopen

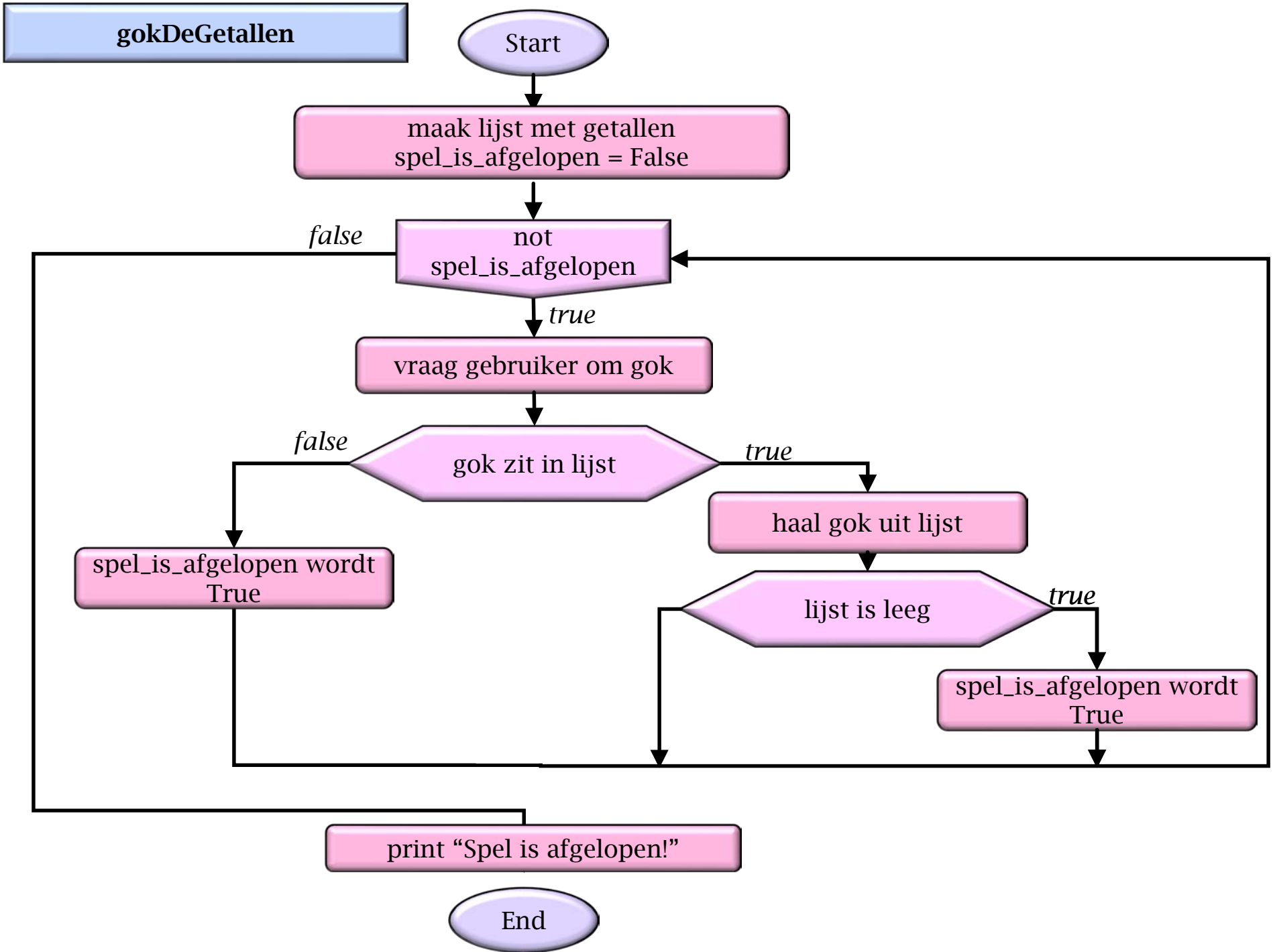
Welke gegevens heb je nodig?

  - List: woorden_lijst
  - Int: gebruikers_gok
  - Boolean: spel_is_afgelopen
- Extra acties:
  - Als gok in getallen_lijst, dan uit lijst verwijderen
  - Als gok niet in getallen_lijst, dan spel_is_afgelopen op True zetten

Welke data en hoe heten ze??

- Lijst met getallen:
  - `geheime_getallenLijst`
- Getal dat gebruiker gokt:
  - `gebruikers_gok`
- Boolean vlag om aan te geven of spel al is afgelopen of niet:
  - `spel_is_afgelopen`

Maak een lijst met een paar getallen er in
Zolang het spel niet is afgelopen:
    Vraag de gebruiker om een getal in te typen
    Als het getal voorkomt in de lijst
        haal je het getal eruit
        check of lijst leeg: spel is afgelopen
    Als het getal niet voorkomt in de lijst, is het spel afgelopen.


gokDeGetallen
Start
maak lijst met getallen
spel_is_afgelopen = False
false
not
spel_is_afgelopen
true
vraag gebruiker om gok
false
gok zit in lijst
true
haal gok uit lijst
spel_is_afgelopen wordt
True
lijst is leeg
true
spel_is_afgelopen wordt
True
print “Spel is afgelopen!”
End

Spel: Gok De Getallen

- Wat voor gegevens moet je opslaan?
  - Datatype (bv. String, Int, Boolean, List)
  - Geschikte naam voor variabele / constant

Voor dit spel, bijvoorbeeld:

- Boolean spel_is_afgelopen
- List getallen_lijst
- Integer: gebruikers_gok

Spel: Gok De Getallen

Het begin van een oplossing:

```
getallen_lijst = [1,5,-8, 0]

spel_is_afgelopen = False  #bij begin: spel nog niet afgelopen

while not spel_is_afgelopen:
  gok = int( input( "Doe een gok:" )) #vraag gebruiker om een gok
  if gok in getallen_lijst:
        getallen_lijst.remove(gok)
        print(getallen_lijst)  #druk af om te testen
  else:
        spel_is_afgelopen = True
        print("Spel is afgelopen")
```

Galgje: welke stappen en welk data

- Speel in tweetallen het spel Galgje op papier.
- 2 rollen:
  1. De computer (bv. bedenk een woord, vraag letter..)
  2. De speler (bv. gok woord)
- Noteer op papier:
  - Wat doet de computer (welke stappen)?
  - Wat onthoud de computer?
  - Let op: laat de speler niks voor je onthouden, want die kan de spel dan manipuleren
- Ontwerp:
  - Wat voor data sla je op (naam + type)
  - Verloop van programma (stroomdiagram)

PO

Goed programmeren gaat **niet alleen** om een werkende **oplossing…**

maar juist om een **degelijke aanpak en oplossing.**

Laat zien wat je geleerd hebt!

Ga gestructureerd te werk, bedenk uit welke losse onderdelen je programma bestaat en pak ze los van elkaar aan.

PO uitdaging

- Maak het spel Galgje
- Computer tegen een speler
- Hoe verder je komt met uitbreidingen, hoe hoger je cijfer!

Beoordeling:

Je **proces** wordt beoordeeld op:	
Voortgang *Plan/Ontwerp*	Je levert je programma op tijd in (als .zip in MontiPlaza). Je werkt zelfstandig en lost zelf (of met andere leerlingen) fouten in de code op. Een plan/ontwerp is vooraf gemaakt: Stroomdiagram + overzicht van dataopslag(bv. String gebruikersgok). Je ontwerp maakt duidelijk hoe en welke afzonderlijke brokken code je in welke volgorde aan gaat werken/testen (zie **Deel A**).
Je **product** wordt beoordeeld op:	
Volledigheid & originaliteit	Alle onderdelen zijn (op de juiste wijze) ingeleverd. Hoe verder je komt qua functionaliteit (**Deel C:** uitbreidingen, originele invulling), hoe hoger je cijfer. Je uitbreidingen staan kort beschreven in je verslag.
Correctheid *Stijl*	Code werkt precies zoals verwacht (volgens omschrijving **Deel B**). Je programma is robuust en geeft duidelijke foutmeldingen bij onvoorziene omstandigheden (bv. ongeldige invoer). In je verslag staat wat wel/(nog)niet goed werkt. Code is makkelijk te lezen en begrijpen, voorzien van uitstekende commentaar en logische naamgeving (conventies).
Constructie	De kwaliteit van de code: het is duidelijk, efficiënt, elegant, logisch en goed gestructureerd. Er wordt gebruik gemaakt van loops, condities. Het verschil tussen lokale en globale variabelen is helder en volgens de geleerde conventies. Er wordt gebruik gemaakt van functies en parameters. Functies staan bij elkaar, bovenaan de code. Gebruik van 'harde' waardes en globale variabelen worden zo veel mogelijk vermeden.

PO inleveren

- Inleveren op: 2 november om 23:59 (eerder mag ook!)
- Als **.zip** via Montiplaza

- Verslag:
  - (een foto van) je ontwerp/plan en/of stroomdiagram + overzicht van wat voor soort data opgeslagen wordt (bv. String gebruikersgok)
  - Korte uitleg van wat wel/niet werkt, en welke uitbreidingen je hebt toegevoegd.
- Je python code.

PO

- PO omschrijving staat in de planner
- Lees goed wat van je verwacht wordt
  - Wat **en hoe** je moet inleveren
  - Waar je op beoordeeld wordt
  - Hoe je het handig aanpakt
  - Waar je hoge cijfers voor kunt halen
- Werk in tweetallen (kies het liefst iemand met zelfde begeleidingsuur!)
- Maak samen duidelijk afspraken over wie, wat, wanneer gaat doen.

Programmeeromgeving

- Pycharm Edu versie
  - Gratis te downloaden
  - Op school PC’s al geinstalleerd
  - Handleiding opzetten omgeving op ‘onze’ website

Volgende week

- PTA toets op papier
  - Max 70 min (ook voor dyslectici)
  - Zodra je klaar bent mag je aan PO beginnen
- Tips:
  - Maak de toetsvoorbereiding op papier
  - Loop je vast? Bekijk de theorie opnieuw!
  - Typ je antwoorden in op de computer (bv. onderaan planningspagina)