Eerste huiswerkopgave Complexiteit 2023
Inleveren: vóór hoorcollege 4 (dinsdag 28 februari 09u00)
Hoe: PDF op Brightspace
Maak de uitwerking in LaTeX!

Geef een *duidelijke* toelichting/uitleg bij al je antwoorden!

Hieronder staat het algoritme voor **Insertion sort**. Gegeven een array met $n \geq 2$ gehele getallen op plekken 1 t/m n, sorteert het algoritme het oplopend. Voor de volledigheid: het tweede deel van de test op regel 4 ($A[j] > x$) wordt alleen uitgevoerd als het eerste deel ($j > 0$) waar is.

```
1 for i := 2 to n do
2 |   x := A[i];
3 |   j := i − 1;
4 |   while j > 0 and A[j] > x do
5 |   |   A[j + 1] := A[j];
6 |   |   j := j − 1;
7 |   od
8 |   A[j + 1] := x;
9 od
```

a. (7 punten) Toon aan dat de vergelijking $A[j] > x$ op regel 4 maatgevend is voor de complexiteit van dit algoritme. Doe dit door te laten zien dat elke andere regel en het eerste deel van de test op regel 4 ($j > 0$) qua orde van grootte hooguit even vaak gebeuren als deze vergelijking.

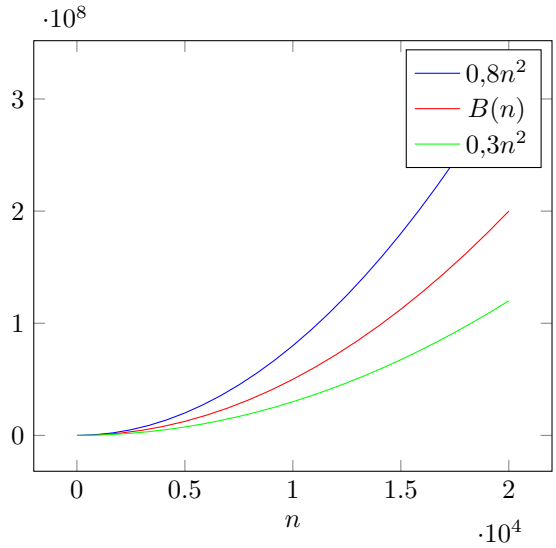
b. (5 punten) Hoeveel arrayvergelijkingen doet dit algoritme in het *beste* geval (voor algemene n) en voor welk(e) invoerrijtje(s) gebeurt dit? Geef *alle* gevallen en leg op basis van het algoritme uit hoe je aan je antwoord komt.

c. (7 punten) Hoeveel arrayvergelijkingen doet dit algoritme in het *slechtste* geval en voor welk(e) invoerrijtje(s) gebeurt dit? Geef *alle* gevallen en leg op basis van het algoritme uit hoe je aan je antwoord komt.

Op `https://liacs.leidenuniv.nl/~edixhovenlj/complexiteit/huiswerk/sorteren.zip` vind je een implementatie van Insertion sort en zes andere sorteeralgoritmen.

- Compileer de code met de bijgeleverde makefile.
- Voer de code uit met `./smurf 10 1000` om 10 keer een willekeurig rijtje van lengte 1000 te genereren en dat met elk algoritme te sorteren.
- Het programma geeft standaard als uitvoer per algoritme het aantal vergelijkingen dat het minimaal, maximaal en gemiddeld heeft moeten doen.

Op de volgende pagina vind je een plot van het aantal vergelijkingen $B(n)$ dat Bubblesort gemiddeld doet om willekeurige rijtjes te sorteren van lengte n, en van de functies $0{,}3n^2$ en $0{,}8n^2$. Op basis van de grafiek is het aannemelijk dat $B(n) \in O(0{,}8n^2) = O(n^2)$ en dat $B(n) \in \Omega(0{,}3n^2) = \Omega(n^2)$ — en daarmee dat $B(n) \in \Theta(n^2)$.



d. (24 punten) Benader op soortgelijke wijze de average case-complexiteit van de andere zes implementaties (Insertion sort, Mergesort, Quicksort, Shellsort, Shellsort (Hibbard) en Heapsort). Verdere instructies:

- Plot minstens twintig datapunten. Spreid ze (enigszins) evenredig om een (enigszins) nette plot te krijgen.
- Zorg ervoor dat je laatste datapunt minstens 100 000 000 (10^8) vergelijkingen doet.
- Plot twee andere functies in dezelfde grafiek: één grotere en één kleinere. Houd je constante factoren (0,3 en 0,8 voor Bubblesort) eenvoudig, en maak ze niet kleiner dan 0,1 en niet groter dan 5. Probeer je geplotte datapunten netjes in te klemmen om zo een precieze orde van grootte te kunnen afschatten.
- Vermeld duidelijk je gevonden orde van grootte.

Tips:

- De LaTeX-template voor deze opdracht (te vinden op de website) bevat de grafiek voor Bubblesort; deze kan je gebruiken als voorbeeld. Zie voor meer details bijvoorbeeld de documentatie van pgfplots: `https://ctan.org/pkg/pgfplots`.
- Het staat je vrij om het programma aan te passen. Met wat aanpassingen in de functies `main` en `run` kan je het genereren van je datapunten (grotendeels) automatiseren.

e. (7 punten) Kijk ook hoeveel vergelijkingen alle implementaties doen voor de rijtjes uit opgaven **b** en **c**, voor n minstens 10 000. Hiervoor zul je het programma zo moeten aanpassen dat het deze rijtjes gebruikt in plaats van dat het willekeurige genereert. Beschrijf je bevindingen en eventuele afwijkingen met je bevindingen uit opgave **d**.