ZBIÓR CANTORA.

M. G. T. P.

Wydział Fizyki, Matematyki i Informatyki
Politechnika Krakowska im. Tadeusza Kościuszki

8 luty 2019

Plan.

Wstęp.

Georg Cantor (1845-1918) – niemiecki matematyk. Studiował w Darmstadt, Zurychu i Getyndze. Doktorat obronił w 1867 roku w Berlinie. Uczył w berlińskim gimnazjum i ponad trzydzieści lat był profesorem uniwersytetu w Halle. Cantor miał znaczący udział w tworzeniu podwalin nowoczesnej matematyki. W szczególności uchodzi za twórcę teorii mnogości.

Wstęp.

Definicja

Zbiór Cantora jest to podzbiór prostej rzeczywistej opisany w 1883 roku przez niemieckiego matematyka Georga Cantora. Zbiór ten odkrył w 1875 roku Henry Smith. Zbiór Cantora jest najprostszym przykładem fraktala.

Definicja

Fraktal w znaczeniu potocznym oznacza zwykle obiekt samopodobny (tzn. taki, którego części są podobne do całości) albo „nieskończenie złożony" (ukazujący coraz bardziej złożone detale w dowolnie wielkim powiększeniu).

Konstrukcja zbioru Cantora.

Klasyczny zbiór Cantora definiujemy jako podzbiór przedziału domkniętego $C_0 = [0, 1]$ liczb rzeczywistych wyznaczony przez następującą konstrukcję. Indukcyjnie wybieramy zstępujący ciąg zbiorów domkniętych $(C_0, C_1, C_2, ...)$, takich że zbiór C_n jest sumą 2^n rozłącznych odcinków domkniętych. Przypuśćmy, że wyznaczyliśmy już zbiór C_n tak, że jest sumą 2^n rozłącznych odcinków domkniętych. Każdy z 2^n odcinków tworzących ten zbiór dzielimy na 3 rozłączne odcinki równej długości z których środkowy odcinek jest otwarty, a odcinki skrajne są domknięte. Wyrzucamy ze zbioru C_n wszystkie środkowe odcinki otwarte kładąc $C_{n+1} = C_n \setminus (I_1 \cup \ldots \cup I_{2^n})$. Można sprawdzić, że zbiór C_{n+1} jest sumą 2^{n+1} rozłącznych odcinków domkniętych. Po zakończeniu procesu indukcyjnego, gdy ciąg $(C_0, C_1, C_2, ...)$ jest wyznaczony, definiujemy zbiór Cantora jako część wspólną tego ciągu.

Program.

```
can :: [(Bool, Int)] -> [(Bool, Int)]

can x =

    let s (l, k)

        | l  1 < k = [(True, p), (False, p), (True, p)]

        | otherwise = [(l, k)]

        where

          p = quot k 3

    in x >>= s
```

Program.

```
canL :: Int -> String

canL k =

    unlines $ pokCan <$> take k (iterate can [(True, 3 ∧(k − 1))])

pokCan :: [(Bool, Int)] -> String

pokCan =

    concatMap

        (\(l, k)− >

            replicate
```

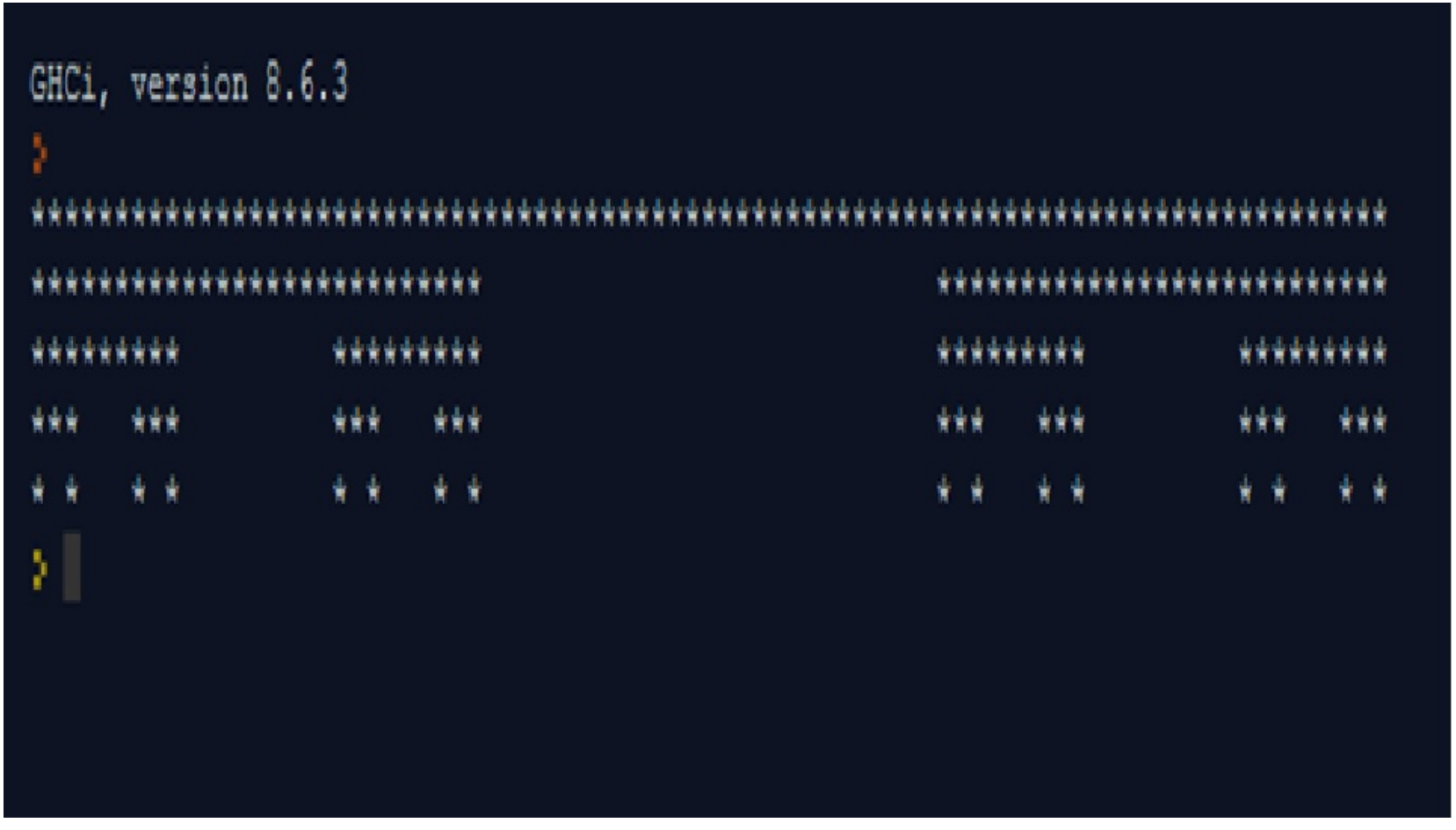
Program.

```
            k
             (if l
                then '*'
                else ' '))

main :: IO ()

main = putStrLn $ canL 4
```

Wyniki.



Bibliografia.


J. Kudrewicz.
Fraktale i chaos.
WNT 2007