# Wizualizacja danych - Gnuplot

dr hab. Bożena Woźna-Szcześniak

Akademia im. Jan Długosza

bwozna@gmail.com

Laboratorium 4

# O czym dziś będzie mowa

- Zarządzanie opcjami Gnuplota
- Format pliku danych
- Łańcuchy
- Wyjście tekstowe
- System pomocy

# Zarządzanie opcjami

- Gnuplot ma stosunkowo niewiele poleceń (np. *plot*, *save*, *load*, *print*), ale wyposażony jest w dużą liczbę opcji.
- Opcje używane są do kontroli wszystkiego, od formatu przecinka do nazwa pliku wyjściowego.
- Do manipulowania poszczególnymi opcjami służą następujące polecenia:
  - *show* - wyświetla aktualną wartość opcji
  - *set* - zmienia wartość opcji
  - *unset* - wyłącza konkretną opcję, lub przywraca wartości domyślne
  - *reset* - przywraca wartości domyślne dla wszystkich opcji, za wyjątkiem opcji terminala i wyjścia.

# Zarządzanie opcjami

1. Ustawiamy globalny styl funkcji na kreślenie przy pomocy punktów
2. Wyświetlamy bieżące ustawienia stylu funkcji
3. Przywracamy styl funkcji do wartości domyślnych.
4. Wyświetlamy bieżące ustawienia stylu funkcji

```
gnuplot> set style function points
gnuplot> show style function
Functions are plotted with points
gnuplot> unset style function
gnuplot> show style function
Functions are plotted with lines
```

# Zarządzanie opcjami

- Polecenie *show* można również stosować do tego, aby wyświetlić wszystkie rodzaje informacji o stanie wewnętrznym `Gnuplota`, np.

```
gnuplot> show variables
User and default variables:
pi = 3.14159265358979
NaN = NaN
GNUTERM = "aqua"
```

# Zarządzanie opcjami

Zmienne zdefiniowane przez użytkownika:

```
gnuplot> tysiac = 1000
gnuplot> show variables
  User and default variables:
  pi = 3.14159265358979
  NaN = NaN
  GNUTERM = "aqua"
  tysiac = 1000
```

# Zarządzanie opcjami

Funkcje zdefiniowane przez użytkownika:

```
gnuplot> g(x) = x*x
gnuplot> f(x) = x**3
gnuplot> plot g(x), f(x)
gnuplot> show functions
  User-Defined Functions:
  g(x) = x*x
  f(x) = x**3
```

# Zarządzanie opcjami

```
gnuplot> show version long
   G N U P L O T
   Version 4.6 patchlevel 0    last modified 2012-03-04
   Build System: Linux i686
   Copyright (C) 1986-1993, 1998, 2004, 2007-2012
   Thomas Williams, Colin Kelley and many others
   gnuplot home:     http://www.gnuplot.info
   faq, bugs, etc:   type "help FAQ"
   immediate help:   type "help"  (plot window: hit 'h')
Compile options:
-READLINE  +LIBEDITLINE  +HISTORY
-BACKWARDS_COMPATIBILITY  +BINARY_DATA
+GD_PNG  +GD_JPEG  +GD_TTF  +GD_GIF  +ANIMATION
-USE_CWDRC  +X11  +X11_POLYGON  +MULTIBYTE  +X11_EXTERNAL +
 USE_MOUSE  +HIDDEN3D_QUADTREE
+DATASTRINGS  +HISTOGRAMS  +OBJECTS  +STRINGVARS  +MACROS
+IMAGE  +USER_LINETYPES +STATS
GNUPLOT_DRIVER_DIR = "/usr/lib/gnuplot"
GNUPLOT_PS_DIR     = "/usr/share/gnuplot/gnuplot/4.6/PostScript"
HELPFILE           = "/usr/share/gnuplot/gnuplot.gih"
```

# Zarządzanie opcjami

- *show all* - pokazuje wszystkie możliwe opcje wraz z ich wartościami domyślnymi. Pokazuje również wartości zmiennych *variables* i functions oraz podaje informacje, które można uzyskać poleceniem *show version long*.
- Będzie to długa..... lista :)

# Pliki z danymi - dozwolone formaty i opcje

- Pliki danych dla `Gnuplota` to pliki tekstowe ASCII zawierające dane w kolumnach, które oddzielone są białymi znakami.
- Możemy wybierać dwie dowolne kolumny i narysować dla nich wykres stosując dyrektywę `using`.
- Gnuplot może czytać zarówno liczby całkowite, liczby zmiennoprzecinkowe, a także liczby w notacji naukowej (tj. 35100 = 3.51e4, -0.0001 = -1e-4, 35100 = 3.51E4, -0.0001 = -1E-4)

# Pliki z danymi - dozwolone formaty i opcje

- Domyślnie, jeśli w pliku z danymi linia zaczyna się od znaku #, to linia ta jest traktowana jako komentarz.
- Polecenie `set datafile commentschar ["str:chars"]` pozwala na ustalenie nowego łańcucha znaków, który będzie rozpoczynać linie komentarza, np.

```
set datafile commentschar "!"
```

```
1970 1
1974 4
1979 4
# Komentarz jest krzyzykiem
1971 3
1973 6
1978 5
1980 2
```

```
1970 1
1974 4
1979 4
! Komentarz jest
! wykrzyknikiem
1971 3
1973 6
1978 5
1980 2
```

## Pliki z danymi - dozwolone formaty i opcje

- Domyślnie pola (kolumny) są oddzielone od siebie odstępami, które wprowadzane są przy pomocy dowolnej liczby znaków spacji lub tabulacji.
- Można zmienić separator pól przy użyciu polecenia: `set datafile separator [ "str:char" | whitespace ]`, np.:

```
set datafile separator "="
```

```
! Separatorem jest teraz znak rowna sie
1970=1
1974=4
1979=4
1971=3
1973=6
1978=5
1980=2
```

# Pliki z danymi - dozwolone formaty i opcje

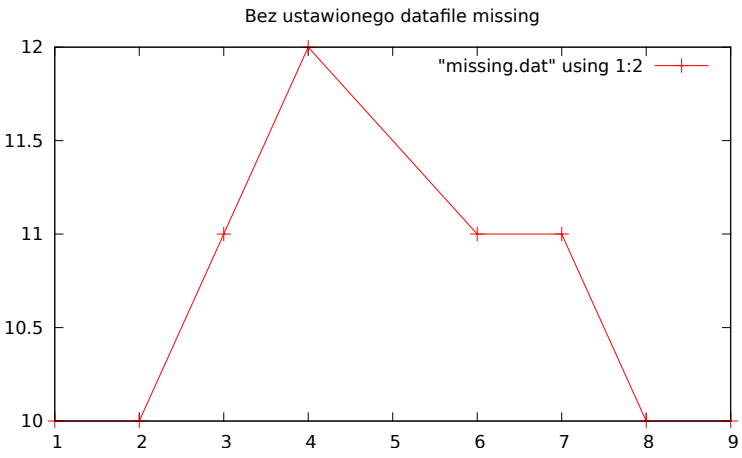
- Znaki separatora nie są interpretowane jako separatory, gdy znajdują się wewnątrz cudzysłowów.
- Ciągi znaków znajdujące się w cudzysłowie zawsze interpretowane są jako dane z jednej kolumny.
- Tylko jeden znak może być zdefiniowany jako separator kolumn w danym czasie; ta uwaga nie dotyczy białych znaków.
- Polecenie `set datafile separator whitespace` lub `set datafile separator` przywraca domyślne ustawienia separatorów.s

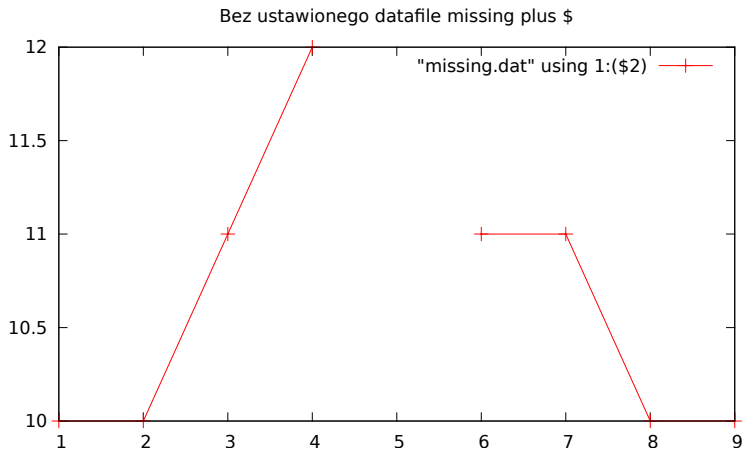
# Pliki z danymi - dozwolone formaty i opcje

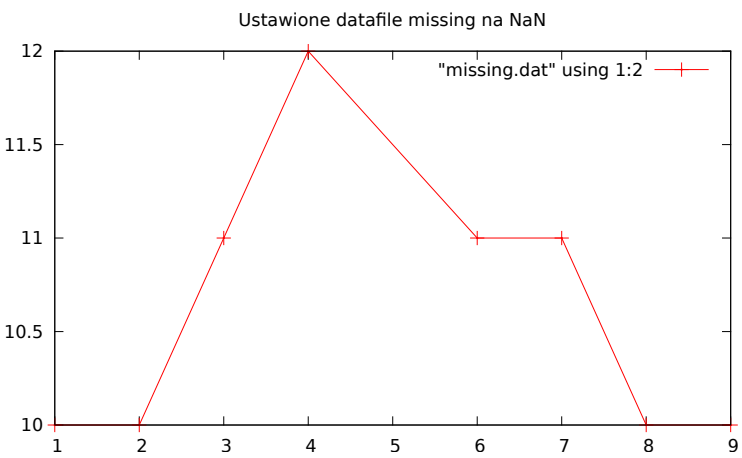
- Polecenie `set datafile missing ["str:str"]` pozwala na zdefiniowanie łańcucha, który oznaczać będzie brakujące daną, np. `set datafile missing "NaN"`

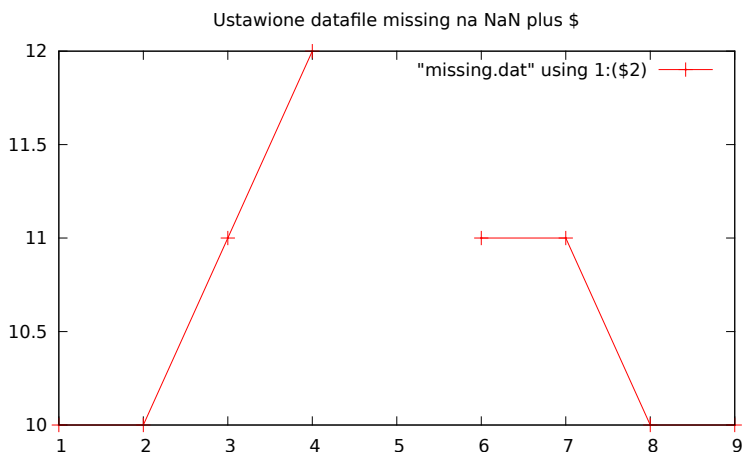
```
set terminal pdf
set output "NaN1.pdf"
set title "Bez ustawionego datafile missing"
plot "missing.dat" using 1:2 with linespoints

set output "NaN2.pdf"
set title "Bez ustawionego datafile missing plus $"
plot "missing.dat" using 1:($2) with linespoints

set datafile missing "NaN"
set output "NaN3.pdf"
set title "Ustawione datafile missing na NaN"
plot "missing.dat" using 1:2 with linespoints

set output "NaN4.pdf"
set title "Ustawione datafile missing na NaN plus $"
plot "missing.dat" using 1:($2) with linespoints
```

# Pliki z danymi - dozwolone formaty i opcje



# Pliki z danymi - dozwolone formaty i opcje

```
gnuplot> show datafile
   No missing data string set for datafile
   datafile fields separated by whitespace
   Comments chars are "#"
```

# Pliki z danymi - dozwolone formaty i opcje

```
gnuplot> set datafile missing "NaN"
gnuplot> show datafile
   "NaN" in datafile is interpreted
       as missing value
   datafile fields separated by whitespace
   Comments chars are "#"
```

# Łańcuchy

- Przetwarzanie łańcuchów jest obsługiwane przez Gnuplota od wersji 4.2.
- Aby się upewnić, czy nasz Gnuplot obsługuje łańcuchy, wystarczy widać polecenie `show version long` i sprawdzić czy ustawione są opcje: `+DATASTRINGS` i `+STRINGVARS`.
- Stałe łańcuchowe w Gnuplocie można przypisać do zmiennej, ale muszą być ujęte w cudzysłowy, albo pojedyncze (apostrofy) albo podwójne.
- Różnica pomiędzy cudzysłowami jest taka, że znaki sterujące (np. `\n`) są interpretowane jako znaki sterujące tylko w łańcuchu ujętym w cudzysłowy podwójne. Znaki sterujące w łańcuchu ujętym w cudzysłowy pojedyncze (apostrofy) są traktowane dosłownie, czyli nie są interpretowane.

# Łańcuchy

- Łańcuch ujęty w cudzysłowy może zawierać apostrofy
- Aby uzyskać cudzysłów wewnątrz cudzysłowia, należy znak cudzysłowia poprzedzić znakiem backslash.
- W łańcuchu ujętym w apostrofy, możemy uzyskać apostrof przez podwojenie go.

```
gnuplot> a = 'This is a string.'
gnuplot> b = "Pierwsza linia \n Druga linia"
gnuplot> c = "Podwojny cudzyslow \" ."
gnuplot> d = 'Pojedynczy cudzyslow ''.'
gnuplot> print a
This is a string.
gnuplot> print b
Pierwsza linia
 Druga linia
gnuplot> print c
Podwojny cudzyslow " .
gnuplot> print d
Pojedynczy cudzyslow '.
```

# Łańcuchy

- Łańcuchy mogą być przypisywane do zmiennych, tak jak wartości liczbowe.
- Łańcuchy mogą zostać skonwertowane do liczby, jeśli to możliwe
- Tylko liczby całkowite mogą zostać skonwertowane do łańcucha.

```
gnuplot> x = '3.14'
gnuplot> y = 2+x
gnuplot> print x
3.14
gnuplot> print y
5.14
gnuplot> a = 4
gnuplot> b = 'cztery'.a
gnuplot> print a
4
gnuplot> print b
cztery4
```

# Łańcuchy

- Istnieją trzy operatory działające na łańcuchach i parę funkcji. Pierwszy z operatorów to operator "kropka", czyli konkatenacja.

```
gnuplot> a = 'baz'
gnuplot> b = 'ar'
gnuplot> c = a.b
gnuplot> print c
bazar
```

# Łańcuchy

- Pozostałe operatory, to operatory porównania:
  - `eq` (równe) - zwraca *prawda (true)*, jeśli oba jego argumenty są równe
  - `ne` (różne) - zwraca *prawda (true)*, jeśli oba jego argumenty są różne.

```
gnuplot> a = 'baz'
gnuplot> c = a eq 'baz' ? 'rowne' : 'rozne'
gnuplot> print c
rowne
```

# Łańcuchy

- Łańcuchy mogą być indeksowane tak jak tablice.
- Pierwszy znak w łańcuchu ma indeks 1.
- Jeśli zostawimy pusty indeks początkowy (końcowy), to domyślnym będzie indeks nr 1 (indeks oznaczający koniec łańcucha).

```
gnuplot> a = "Gnuplot"
gnuplot> b = a[2:4]
gnuplot> c = a[4:]
gnuplot> print b
nup
gnuplot> print c
plot
```

# Łańcuchy

| Funkcja | Opis |
|---|---|
| `strlen("str")` | Zwraca długość napisu *str* |
| `substr("str", i, j )` | Równoważne *str[i:j].* |
| `strstrt("str", "key")` | Zwraca indeks pierwszego znaku łańcucha *key* w łańcuchu *str*, lub zero jeśli nie znaleziono. |
| `words("str")` | Zwraca liczbę znalezionych tokenów |

```
gnuplot> print words("Raz dwa trzy.
                      Raz, dwa, trzy !")
7
```

# Łańcuchy

| Funkcja | Opis |
|---|---|
| `word( "str", n )` | zwraca n-ty token |
| `sprintf( "format", ...)` | sformatowany łańcuch znaków; równoważna *sprintf()* z C |
| `system( "str" )` | wykonuje polecenie powłoki. |

```
gnuplot> print word("Raz dwa trzy.
                          Raz, dwa, trzy !", 2)
dwa
gnuplot> system ("ls")
Gnuplot04.tex    jumbled2.dat   jumbled3.dat
NaN1.pdf  jumbled4.dat   NaN2.pdf  logo_AJD.pdf
Gnuplot04.pdf  NaN3.pdf  missing.dat
NaN4.pdf  missing1.dat jumbled1.dat  skryp1.p
```

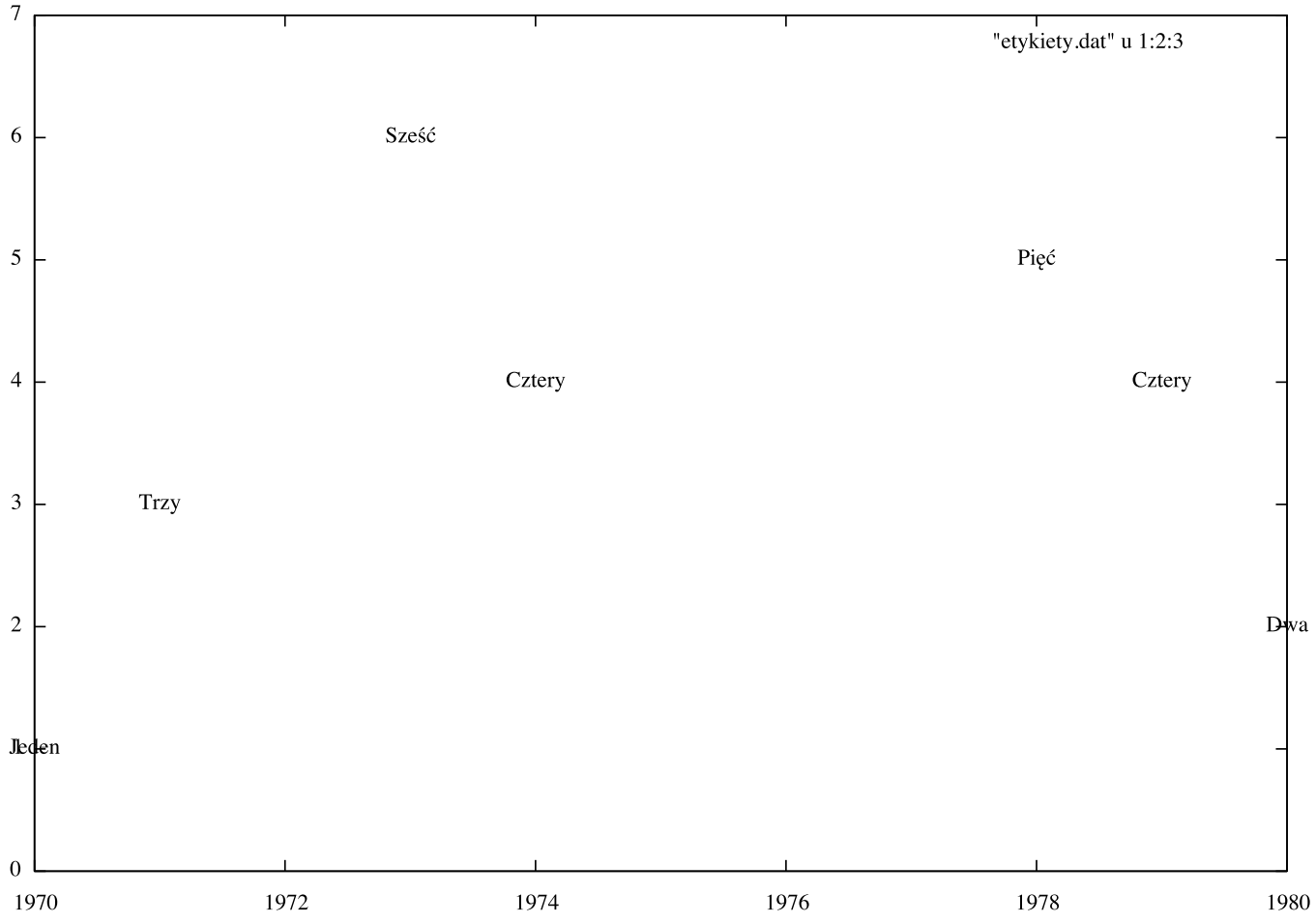
# Łańcuchy

Załóżmy, że mamy następujący plik "etykiety.dat"

```
1970 1  Jeden
1974 4  Cztery
1979 4  Cztery
1971 3  Trzy
1973 6  Sześć
1978 5  Pięć
1980 2  Dwa
```

# Łańcuchy

```
gnuplot> plot [][0:7] "etykiety.dat"
                  u 1:2:3 with labels
```



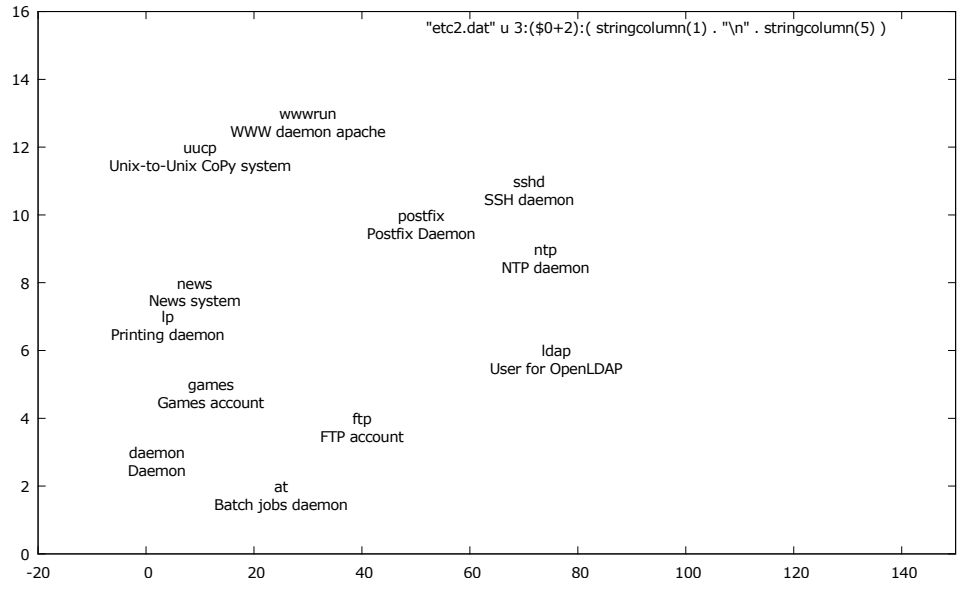
# Łańcuchy

Załóżmy, że mamy następujący plik "etc.dat"

```
at:x:25:25:Batch jobs daemon:/var/spool/atjobs:/bin/bash
daemon:x:2:2:Daemon:/sbin:/bin/bash
ftp:x:40:49:FTP account:/srv/ftp:/bin/bash
games:x:12:100:Games account:/var/games:/bin/bash
ldap:x:76:70:User for OpenLDAP:/var/lib/ldap:/bin/bash
lp:x:4:7:Printing daemon:/var/spool/lpd:/bin/bash
mail:x:8:12:Mailer daemon:/var/spool/clientmqueue:/bin/false
man:x:13:62:Manual pages viewer:/var/cache/man:/bin/bash
mysql:x:60:108:MySQL database admin:/var/lib/mysql:/bin/false
news:x:9:13:News system:/etc/news:/bin/bash
ntp:x:74:103:NTP daemon:/var/lib/ntp:/bin/false
postfix:x:51:51:Postfix Daemon:/var/spool/postfix:/bin/false
sshd:x:71:65:SSH daemon:/var/lib/sshd:/bin/false
uucp:x:10:14:Unix-to-Unix CoPy system:/etc/uucp:/bin/bash
wwwrun:x:30:8:WWW daemon apache:/var/lib/wwwrun:/bin/false
```

# Łańcuchy

```
set terminal pdf font ’Verdana,8’
set output "etc.pdf"
set datafile separator ’:’
set datafile commentschar "m"
# Pseudokoluma 0 zawiera numer linii w bieżącym zbiorze
plot [-20:150][0:16] "etc.dat" u
3:($0+2):( stringcolumn(1) . "\n" . stringcolumn(5) ) w labels
```



# Komenda `print`

```
gnuplot> print sin(1.5*pi)
-1.0
gnuplot> print "The value of pi is: ", pi
The value of pi is: 3.14159265358979
```

Urządzenie, do którego `print` będzie wysyłać swoje wyjście może zostać zmieniony przez komende `set print`.

# **Komenda** `print`

- Domyślnie komenda `print` wysyła dane na standardowe wyjście błędów; zwykle jest to terminal, z którego gnuplot został uruchamiany.
  `set print`
- Wyjście może być przekierowane albo na standardowe wyjście (przy użyciu specjalnego pliku "-") lub zwykłego pliku (przez podanie nazwy pliku).
  `set print "-"` lub
  `set print "str:filename" [ append ]`
- Każde wywołanie `set print` tworzy nowy plik chyba, że dodatkowe słowo kluczowe `append` zostało zastosowane.

# **Komenda** `set table`

- Komenda `set table` daje nam dostęp do danych, które tworzy wykres w postaci tekstu, czyli za pomocą `set table`, możemy uzyskać wartości wszystkich punktów przedstawionych na wykresie jako wyrażenia numeryczne.
- Aby wygenerować dane wyjściowe w postaci pliku tekstowegp zawierającego liczby, a nie w postaci graficznej, należy zastosować polecnie:
  ```
  set table ["str:filename"]
  ```
- Aby przywrócić generowanie wykresów należy wydać komendę:
  ```
  unset table
  ```

# **Komenda** `set table`

Zestaw komend:

```
gnuplot> set table "sinus.txt"
gnuplot> plot sin(x)
```

wygeneruje plik *"sinus.txt"*:

```
# Curve 0 of 1, 100 points
# Curve title: "sin(x)"
# x y type
-10  0.544021  i
-9.79798  0.364599  i
-9.59596  0.170347  i
-9.39394 -0.0308337  i
-9.19192 -0.23076  i
-8.9899 -0.421301  i
.........
 8.9899  0.421301  i
 9.19192  0.23076  i
 9.39394  0.0308337  i
 9.59596 -0.170347  i
 9.79798 -0.364599  i
 10 -0.544021  i
```

# **Komenda** `set table`

- Trzecia kolumna zawiera znacznik wskazujący, czy dany punkt danych był w zakresie danych lub poza nim:
  - `i` - "w zakresie",
  - `o` - "poza zakresem",
  - `u` - punkt jest niezdefiniowany.

# **Komenda** `help`

```
gnuplot> help plot
‘plot‘ is the primary command for drawing plots with ‘gnuplot‘.
It creates plots of functions and data in many, many ways.
‘plot‘ is used to draw 2D functions and data;
‘splot‘ draws 2D projections of 3D surfaces and data.
‘plot‘ and ‘splot‘ offer many features in common;
see ‘splot‘ for differences. Note specifically that although
the ‘binary <binary list>‘ variation does work for both ‘plot‘
and ‘splot‘, there are small differences between them.
Syntax:
  plot {<ranges>}
       {<iteration>}
       {<function> | {"<datafile>" {datafile-modifiers}}}
       {axes <axes>} {<title-spec>} {with <style>}
       {, {definitions{,}} <function> ...}
```

# Interaktywne wprowadzanie danych

```
gnuplot> plot '-'
input data ('e' ends) > 2
input data ('e' ends) > 4
input data ('e' ends) > 6
input data ('e' ends) > 8
input data ('e' ends) > 10
input data ('e' ends) > (Ctrl+D)
gnuplot>
```