

**Wykonał:** Radosław Smoter

**Grupa:** 14

**Nr:** 27

**Numer zadania:** 2

**Przykład:** 62

**Prowadzący:** Prof. dr hab. inż.  
Volodymyr Samotyy

# **Politechnika Krakowska**

**Wydział Inżynierii Elektrycznej i Komputerowej**

**Sprawozdanie: Wstęp do Programowania**

## Spis treści

Polecenie.....	1
Kod programu.....	2
Wyniki.....	4
Opis Programu.....	8
Wnioski.....	9

## Polecenie

Obliczyć wartości funkcji z wyborem formuły. Wyniki obliczeń zapisać do pliku tekstowego. Narysować wykres  $y(x)$  .

Funkcje:

$$y=1.7-\ln(x^2)+3.2\lg(x^3), x\in(-\infty, 0.6) \text{ ,}$$

$$y=\sqrt{3x+\ln(x-1.2)}, x\in[0.6, 0.7) \text{ ,}$$

$$y=2.7+4x^2+2.3x^3, x\in[0.7, +\infty)$$

Dziedzina:  $x\in[0.5, 0.9]$  .

$\lg(x)$  przyjmuję za  $\log_{10}(x)$  ,

$\ln(x)$  przyjmuję za  $\log_e(x)$  .

## Kod programu

```

/**
 * @file main.c
 * @author Radosław Smoter (radoslaw.smoter@student.pk.edu.pl)
 * @version 0.1
 * @date 2021-12-16

 * @copyright Copyright (c) 2021
 */

/*
 * Calculate given functions with their respectable domains.
 * Results save to a file.
 * Draw a plot for the results.
 */

#include <stdio.h>
#include <math.h>

/* Calculate mathematical function */
double countFirst(double x);
/* Calculate mathematical function */
double countSecond(double x);
/* Calculate mathematical function */
double countThird(double x);

int main(void)
{
    /* Domain min/max*/
    double domain[] = { 0.5, 0.9 };
    /* Domain step */
    double step = 0.002;

    FILE *file = fopen("results.dat", "w");

    /* Ensure file exists */
    if (file != NULL)
        for (double x = domain[0]; x < domain[1]; x += step) {
            /* Step result */
            double y;

            /* If in domain, do respectable function */
            if (x < .6) y = countFirst(x);
            else if (x >= .6 && x < .7) y = countSecond(x);
            else y = countThird(x);

            if ( ! isnan(y))
                fprintf(file, "%f\t%f\n", x, y);

            fclose(file);
        }
    /* Error: File did not open correctly. */
    else {

```

```

        fprintf(stderr, "Error: File did not open correctly.\n");
        return -1;
    }

    return 0;
}

/* If dangerous operations are encountered, verify whether their values
are numbers; otherwise return NAN.*/

double countFirst(double x) {
    double log1 = log(pow(x, 2));
    double log2 = log(pow(x, 3));
    /* Verify whether numbers */
    if (isnan(log1) || isnan(log2)) return NAN;
    return 1.7 - log1 + 3.2 * log2 / log(10);
}

double countSecond(double x) {
    double log1 = log(x - 1.2);

    /* Check if not a number */
    if (isnan(log1)) return NAN;

    double sqrt1 = pow(3 * x + log1, 1/2);

    /* Check if not a number */
    if (isnan(sqrt1)) return NAN;

    return sqrt1;
}

double countThird(double x) {
    return 2.7 + 4 * pow(x, 2) + 2.3 * pow(x, 3);
}

```

## Wyniki

x	y
0.50000	0.19641
0.50200	0.20507
0.50400	0.21369
0.50600	0.22228
0.50800	0.23084
0.51000	0.23936
0.51200	0.24785
0.51400	0.25631
0.51600	0.26473
0.51800	0.27313
0.52000	0.28149
0.52200	0.28981
0.52400	0.29811
0.52600	0.30637
0.52800	0.31460
0.53000	0.32280
0.53200	0.33098
0.53400	0.33911
0.53600	0.34722
0.53800	0.35530
0.54000	0.36335
0.54200	0.37137
0.54400	0.37936
0.54600	0.38732
0.54800	0.39525
0.55000	0.40316
0.55200	0.41103
0.55400	0.41887
0.55600	0.42669
0.55800	0.43448
0.56000	0.44224
0.56200	0.44998
0.56400	0.45768
0.56600	0.46536
0.56800	0.47301
0.57000	0.48064
0.57200	0.48823
0.57400	0.49581
0.57600	0.50335
0.57800	0.51087
0.58000	0.51836
0.58200	0.52583
0.58400	0.53327
0.58600	0.54069
0.58800	0.54808
0.59000	0.55544

0.59200	0.56279
0.59400	0.57010
0.59600	0.57739
0.59800	0.58466
0.70000	5.44890
0.70200	5.46690
0.70400	5.48497
0.70600	5.50310
0.70800	5.52131
0.71000	5.53960
0.71200	5.55795
0.71400	5.57637
0.71600	5.59487
0.71800	5.61343
0.72000	5.63207
0.72200	5.65078
0.72400	5.66956
0.72600	5.68842
0.72800	5.70734
0.73000	5.72634
0.73200	5.74541
0.73400	5.76455
0.73600	5.78377
0.73800	5.80305
0.74000	5.82242
0.74200	5.84185
0.74400	5.86135
0.74600	5.88093
0.74800	5.90059
0.75000	5.92031
0.75200	5.94011
0.75400	5.95998
0.75600	5.97993
0.75800	5.99995
0.76000	6.02004
0.76200	6.04021
0.76400	6.06045
0.76600	6.08077
0.76800	6.10116
0.77000	6.12163
0.77200	6.14217
0.77400	6.16278
0.77600	6.18347
0.77800	6.20423
0.78000	6.22507
0.78200	6.24598
0.78400	6.26697
0.78600	6.28804
0.78800	6.30917
0.79000	6.33039

0.79200	6.35168
0.79400	6.37305
0.79600	6.39449
0.79800	6.41601
0.80000	6.43760
0.80200	6.45927
0.80400	6.48102
0.80600	6.50284
0.80800	6.52474
0.81000	6.54671
0.81200	6.56877
0.81400	6.59090
0.81600	6.61310
0.81800	6.63539
0.82000	6.65775
0.82200	6.68018
0.82400	6.70270
0.82600	6.72529
0.82800	6.74796
0.83000	6.77071
0.83200	6.79354
0.83400	6.81644
0.83600	6.83942
0.83800	6.86248
0.84000	6.88562
0.84200	6.90884
0.84400	6.93213
0.84600	6.95550
0.84800	6.97896
0.85000	7.00249
0.85200	7.02610
0.85400	7.04979
0.85600	7.07355
0.85800	7.09740
0.86000	7.12133
0.86200	7.14534
0.86400	7.16942
0.86600	7.19359
0.86800	7.21783
0.87000	7.24216
0.87200	7.26656
0.87400	7.29105
0.87600	7.31561
0.87800	7.34026
0.88000	7.36499
0.88200	7.38979
0.88400	7.41468
0.88600	7.43965
0.88800	7.46470
0.89000	7.48983



0.89200	7.51504
0.89400	7.54033
0.89600	7.56571
0.89800	7.59116

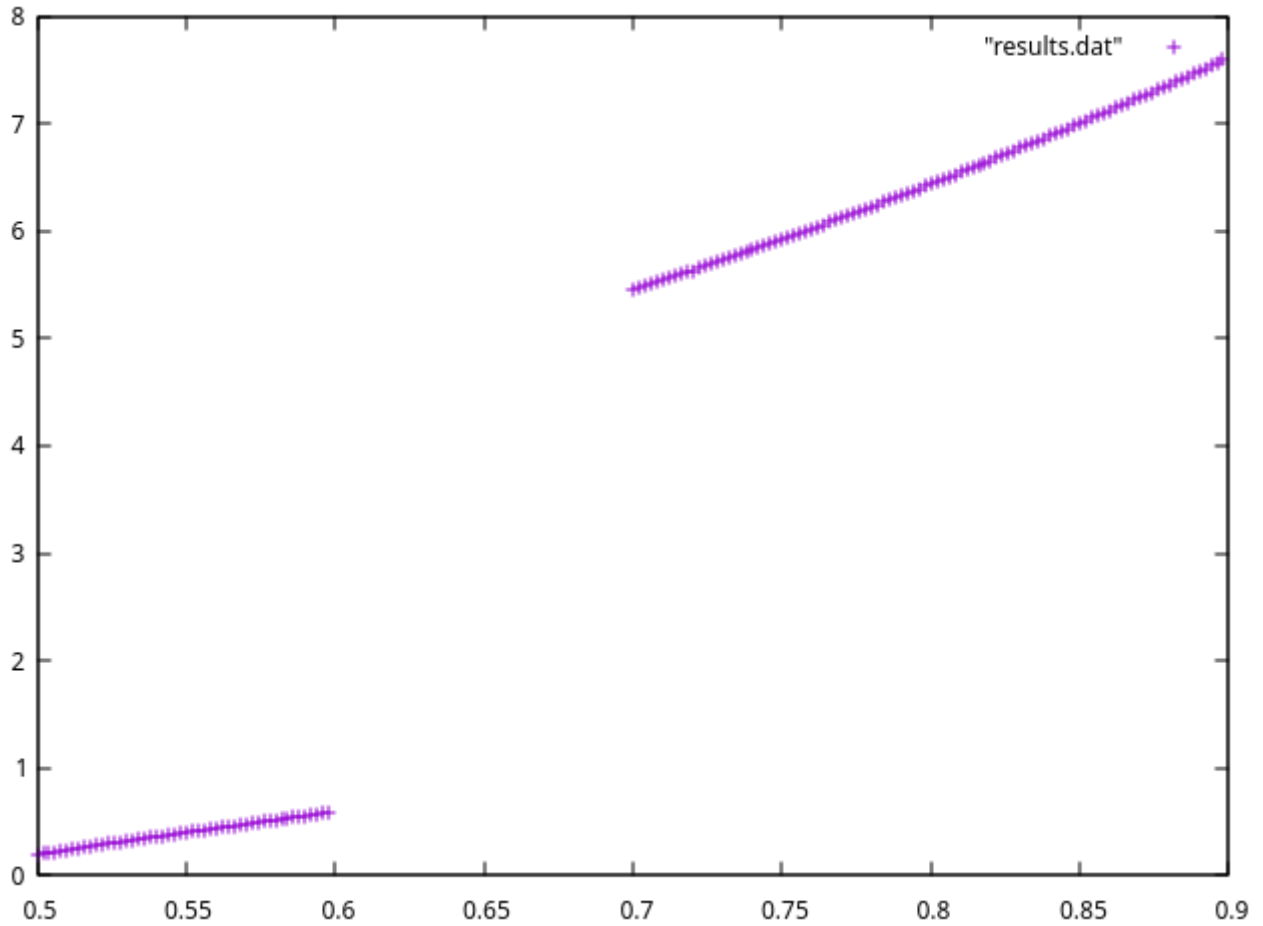


Figura 1: Wykres wartości powyżej, uzyskany programem Gnuplot, poleceniem `<<plot "results.dat">>`

## Opis Programu

Program składa się z funkcji głównej, w której odbywają się operacje na plikach oraz zapis danych. Przebiega w niej również iteracja po określonym kroku (step), gdzie wykonują się funkcje matematyczne zależne od aktualnie dostarczanej wartości  $x$ . Wynik operacji matematycznych zapisywany jest do zmiennej  $y$ , a później zapisywany do pliku „results.dat”.

Operacje matematyczne wykonują się w trzech bliźniaczych funkcjach `countFirst()`, `countSecond()`, `countThird()`. Zawierają one sprawdzenia względem niebezpiecznych operacji (czyli takich, które nie mogą się wykonywać w zadanej dla nich dziedzinie). Jeśli otrzymane wartości nie są poprawnymi liczbami, funkcje te zwracają wartość NAN.

## Wnioski


Wartości zostały obliczone prawidłowo, co łatwo sprawdzić za pomocą różnych programów matematycznych.


Poniżej przedstawiono trzy zadane funkcje w ich odpowiadających przedziałach dziedziny.

$$y = 1.7 - \ln(x^2) + 3.2 \cdot \lg(x^3) \text{ in } (0.5, 0.6)$$

 NATURAL LANGUAGE

 MATH INPUT

 EXTENDED KEYBOARD

 EXAMPLES

 UPLOAD

 RANDOM

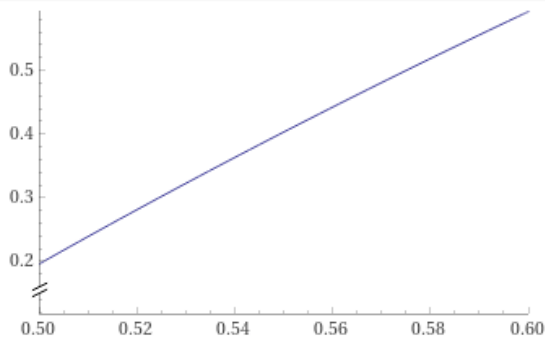
Assuming "lg" is the base 10 logarithm | Use [the natural logarithm](#) or [the base 2 logarithm](#) instead

Input interpretation

plot  $y = 1.7 - \log(x^2) + 3.2 \log_{10}(x^3)$   $x = 0.5 \text{ to } 0.6$

$\log(x)$  is the natural logarithm


Plot




$$y = \sqrt{3 \cdot x + \ln(x - 1.2)} \text{ in } [0.6; 0.7]$$

 NATURAL LANGUAGE

 MATH INPUT

 EXTENDED KEYBOARD

 EXAMPLES

 UPLOAD

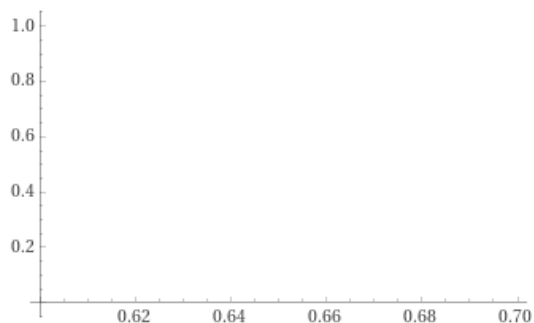
 RANDOM

Input interpretation

plot  $y = \sqrt{3x + \log(x - 1.2)}$   $x = 0.6 \text{ to } 0.7$

$\log(x)$  is the natural logarithm

Plot





$y = 2.7 + 4x^2 + 2.3x^3$  in (0.7; 0.9]



NATURAL LANGUAGE



MATH INPUT



EXTENDED KEYBOARD



EXAMPLES



UPLOAD



RANDOM

Input interpretation

plot

$$y = 2.7 + 4x^2 + 2.3x^3$$

$x = 0.7$  to  $0.9$

Plot

