КПІ ім. Ігоря Сікорського

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт до комп’ютерного практикуму з курсу “Основи програмування”

Прийняв
асистент кафедри ІПІ
Ахаладзе А.Е.
“25” вересня 2024 р.

Виконав
Студент групи ІП-43
Дутов І.А.

Київ 2024

Комп’ютерний практикум №2

Тема: Програмування лінійних алгоритмів

Завдання:

Написати програму, що за введеними сторонами трикутника обчислює його:

- площу;
- периметр;
- висоти;
- бісектриси;
- медіани.

Текст програми:

Якщо ви не хочете копіювати код або бажаєте подивитися тестування, то можна просто завантажити його за посиланням.

Файлова структура

```
/lab2_triangle_props_from_sides
  build
  include
  │ constants.h
  │ triangle.h
  └ utils.h
  src
  │ main.c
  │ triangle.c
  └ utils.c
  test
  unity
  Makefile
  README.md
```

main.c

```c
#include "../include/triangle.h"

void process_arguments(int argc, char *argv[], int *repeat_mode,
                       int *quiet_mode, int *help_mode);
void print_demo();
```

```
void read_triangle_sides(long double sides[SIDE_COUNT],
                         const char *side_labels[SIDE_COUNT],
                         const char *repeat_side_input);
void read_triangle_properties(long double properties[PROPERTY_COUNT],
                              long double a, long double b, long double c);
void print_properties(long double properties[PROPERTY_COUNT],
                      const char *property_labels[PROPERTY_COUNT],
                      long double sides[SIDE_COUNT], unsigned decimal_places);
void print_help();

int main(int argc, char *argv[]) {
  int repeat_mode = 0;
  int quiet_mode = 0;
  int help_mode = 0;

  process_arguments(argc, argv, &repeat_mode, &quiet_mode, &help_mode);

  if (help_mode) {
    print_help();
    return 0;
  }

  long double sides[SIDE_COUNT] = {0};
  long double properties[PROPERTY_COUNT] = {0};

  const char *property_labels[PROPERTY_COUNT] = {
      "Периметр трикутника:",     "Площа трикутника:",
      "Висота до сторони a:",     "Висота до сторони b:",
      "Висота до сторони c:",     "Медіана до сторони a:",
      "Медіана до сторони b:",    "Медіана до сторони c:",
      "Бісектриса до сторони a:", "Бісектриса до сторони b:",
      "Бісектриса до сторони c:"};
  const char *side_labels[SIDE_COUNT] = {"a", "b", "c"};

  const char *repeat_side_input =
      "Просимо Вас вводити лише програмно коректні сторони трикутника.\n\n";
  const char *repeat_decimal_places_input =
      "Просимо Вас вводити лише програмно коректну точність.\n\n";
  unsigned decimal_places = 0;

  do {
    if (!quiet_mode) {
      print_demo();
```

```
    printf("\n");
  }

  // Get sides from user
  do {
    read_triangle_sides(sides, side_labels, repeat_side_input);
  } while (validate_triangle(sides[0], sides[1], sides[2]) != 0);
  printf("\n");

  // Get decimal places from user
  while (read_decimal_places(&decimal_places) != 0) {
    printf("%s", repeat_decimal_places_input);
  }
  printf("\n");

  // Calculating results
  read_triangle_properties(properties, sides[0], sides[1], sides[2]);
  print_properties(properties, property_labels, sides, decimal_places);
  printf("\n");

  if (!quiet_mode) {
    if (!repeat_mode) {
      printf("\nPS: ");
      print_help();
      printf("Дякуємо за використання нашої програми!\n");
      printf("Бажаємо Вам гарного дня!\n");
      printf("\nНатисніть Enter для виходу. ");
      getchar();
    } else {
      printf("\nНатисніть будь-яку клавішу для продовження або Ctrl+C для "
          "виходу.\n\n");
      getchar();
    }
  }
 } while (repeat_mode);

 return 0;
}

void process_arguments(int argc, char *argv[], int *repeat_mode,
                       int *quiet_mode, int *help_mode) {
 for (int i = 1; i < argc; i++) {
   if (strcmp(argv[i], "-q") == 0 || strcmp(argv[i], "--quiet") == 0) {
```

```
      *quiet_mode = 1;
    } else if (strcmp(argv[i], "-r") == 0 || strcmp(argv[i], "--repeat") == 0) {
      *repeat_mode = 1;
    } else if (strcmp(argv[i], "-h") == 0 || strcmp(argv[i], "--help") == 0) {
      *help_mode = 1;
    }
  }
}

void print_demo() {
  printf("Вас вітає Простокутник -- інструмент, що допоможе вам знайти: \n");
  printf("\t- периметр трикутника\n");
  printf("\t- площу трикутника\n");
  printf("\t- висоти трикутника\n");
  printf("\t- медіани трикутника\n");
  printf("\t- бісектриси трикутника\n");

  printf("\nНАШІ ОБМЕЖЕННЯ:\n");
  printf("\t- максимальна довжина сторони трикутника: %Lg\n",
MAX_SIDE_LENGTH);
  printf("\t- максимальна довжина сторони в символах: %u\n",
         NUMBER_LENGTH_LIMIT);
  printf("\t- мінімальна довжина сторони трикутника: %Lg\n", MIN_SIDE_LENGTH);
  printf("\t- максимальна гарантована точність (КІЛЬКІСТЬ знаків після коми) : "
         "%u\n",
         MAX_GUARANTEED_PRECISION);
  printf("\t- мінімальна гарантована точність: %u\n",
MIN_GUARANTEED_PRECISION);
  printf("\t- застосовується автоматична точність: вона збільшується до тих "
         "пір, \n\t\t доки довжини не можна буде розрізнити як рядки\n");
  printf("\t- округлення НЕ застосовується \n");
  printf("\nДля того, щоб вийти з програми передчасно, натисніть Ctrl+C.\n");
}

void read_triangle_sides(long double sides[SIDE_COUNT],
                         const char *side_labels[SIDE_COUNT],
                         const char *repeat_side_input) {
  for (int i = 0; i < SIDE_COUNT; i++) {
    while (read_triangle_side(&sides[i], side_labels[i]) != 0) {
      printf("%s", repeat_side_input);
    }
  }
}
```

```
void read_triangle_properties(long double properties[PROPERTY_COUNT],
                              long double a, long double b, long double c) {
  properties[0] = get_triangle_perimeter(a, b, c);
  properties[1] = get_triangle_area(a, b, c);
  properties[2] = get_triangle_height(a, b, c);
  properties[3] = get_triangle_height(b, a, c);
  properties[4] = get_triangle_height(c, a, b);
  properties[5] = get_triangle_median(a, b, c);
  properties[6] = get_triangle_median(b, a, c);
  properties[7] = get_triangle_median(c, a, b);
  properties[8] = get_triangle_bisector(a, b, c);
  properties[9] = get_triangle_bisector(b, a, c);
  properties[10] = get_triangle_bisector(c, a, b);
}

void print_properties(long double properties[PROPERTY_COUNT],
                      const char *property_labels[PROPERTY_COUNT],
                      long double sides[SIDE_COUNT], unsigned decimal_places) {
  printf("========================================\n");
  printf("РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:\n");
  printf("========================================\n");
  for (int i = 0; i < PROPERTY_COUNT; i++) {
    print_with_additional_precision(property_labels[i], decimal_places,
                                    properties[i], sides);
  }
};

void print_help() {
  printf("Програму можна запускати з деякими параметрами!\n"
         "-q/--quiet:  вимкнути додатковий вивід у програмі\n"
         "-r/--repeat: програма повторюватиметься після виконання\n"
         "-h/--help: вивести цей текст і припинити виконання програми\n");
}
```

triangle.c

```
#include "../include/triangle.h"
#include "../include/utils.h"

// Core Triangle Calculations

long double get_triangle_perimeter(long double a, long double b,
```

```
                              long double c) {
  return (a + b + c);
}

long double get_triangle_area(long double a, long double b, long double c) {
  long double p = get_triangle_perimeter(a, b, c) / 2.0;
  return sqrtl(p * (p - a) * (p - b) * (p - c));
}

long double get_triangle_height(long double side, long double b,
                                long double c) {
  return get_triangle_area(side, b, c) * 2.0 / side;
}

long double get_triangle_median(long double side, long double b,
                                long double c) {
  return sqrtl(2 * b * b + 2 * c * c - side * side) / 2.0;
}

long double get_triangle_bisector(long double side, long double b,
                                  long double c) {
  long double p = get_triangle_perimeter(side, b, c) / 2;
  return 2 * sqrtl(b * c * p * (p - side)) / (b + c);
}

// Triangle validation
int validate_triangle(long double a, long double b, long double c) {
  if (a <= 0) {
    printf("\nПОМИЛКА! Довжина сторони a має бути додатною!\n");
    return 1;
  }
  if (b <= 0) {
    printf("\nПОМИЛКА! Довжина сторони b має бути додатною!\n");
    return 1;
  }

  if (c <= 0) {
    printf("\nПОМИЛКА! Довжина сторони c має бути додатною!\n");
    return 1;
  }

  if (a + b <= c) {
    printf("\nПОМИЛКА! Сума довжин сторін a та b має бути більшою за довжину "
```

```
            "сторони c!\n");
    return 1;
  }

  if (a + c <= b) {
    printf("\nПОМИЛКА! Сума довжин сторін a та c має бути більшою за довжину "
            "сторони b!\n");
    return 1;
  }

  if (b + c <= a) {
    printf("\nПОМИЛКА! Сума довжин сторін b та c має бути більшою за довжину "
            "сторони a!\n");
    return 1;
  }

  return 0;
}

// Input and precision handling functions

int read_triangle_side(long double *side, const char *side_name) {
  char input[NUMBER_LENGTH_LIMIT + 1];
  char *endptr;

  printf("Введіть, будь ласка, довжину сторони %s: ", side_name);

  if (!fgets(input, sizeof(input), stdin)) {
    perror("\nПОМИЛКА! Не вдалося прочитати ввід!\n");
    return 1;
  }

  if (input[strlen(input) - 1] != '\n') {
    printf("\nПОМИЛКА! Довжина сторони %s в символах має бути меншою за %u.\n",
            side_name, NUMBER_LENGTH_LIMIT);
    flush_input_buffer();
    return 1;
  }

  replace_commas_with_dots(input);
  *side = strtold(input, &endptr);

  // Check if conversion failed or extra characters exist
```

```
  if (endptr == input || *endptr != '\n') {
    printf("\nПОМИЛКА! Довжина сторони %s має бути числом і не містити "
           "додаткових символів!\n",
           side_name);
    return 1;
  }

  // Check for side length constraints
  if (*side < 0) {
    printf("\nПОМИЛКА! Довжина сторони %s має бути додатною!\n", side_name);
    return 1;
  }
  if (*side > MAX_SIDE_LENGTH) {
    printf("\nПОМИЛКА! Довжина сторони %s має бути менша-рівна %Lg.\n",
           side_name, MAX_SIDE_LENGTH);
    return 1;
  }
  if (*side < MIN_SIDE_LENGTH - TOLERANCE) {
    printf("\nПОМИЛКА! Довжина сторони %s має бути більша-рівна %Lg.\n",
           side_name, MIN_SIDE_LENGTH);
    return 1;
  }

  return 0;
}

int read_decimal_places(unsigned *decimal_places) {
  printf("Введіть точність від 1 до %u: ", MAX_GUARANTEED_PRECISION);

  char input[NUMBER_LENGTH_LIMIT + 1];
  char *endptr;
  long value;

  if (!fgets(input, sizeof(input), stdin)) {
    printf("\nПОМИЛКА! Не вдалося прочитати ввід!\n");
    return 1;
  }

  if (input[strlen(input) - 1] != '\n') {
    printf(
        "\nПОМИЛКА! Максимальна довжина значення точності в символах складає "
        "%u.\n",
        NUMBER_LENGTH_LIMIT);
```

```
    flush_input_buffer();
    return 1;
  }

  value = strtol(input, &endptr, 10);

  if (endptr == input || *endptr != '\n') {
    printf("\nПОМИЛКА! Точність має бути числом.\n");
    return 1;
  }

  if (value > MAX_GUARANTEED_PRECISION) {
    printf("\nПОМИЛКА! Точність має бути менша-рівна %u.\n",
           MAX_GUARANTEED_PRECISION);
    return 1;
  }

  if (value < MIN_GUARANTEED_PRECISION) {
    printf("\nПОМИЛКА! Точність має бути більша-рівна %u.\n",
           MIN_GUARANTEED_PRECISION);
    return 1;
  }

  *decimal_places = (unsigned)value;
  return 0;
}

// Precision Adjustment Functions

static int numbers_are_equal_by_string(const long double value1,
                                       const long double value2,
                                       unsigned precision) {
  char buffer1[NUMBER_LENGTH_LIMIT], buffer2[NUMBER_LENGTH_LIMIT];
  snprintf(buffer1, sizeof(buffer1), "%.*Lf", precision, value1);
  snprintf(buffer2, sizeof(buffer2), "%.*Lf", precision, value2);
  return strcmp(buffer1, buffer2) == 0;
}

void print_with_additional_precision(const char *text, unsigned int precision,
                                     long double value,
                                     long double sides[SIDE_COUNT]) {
  char formatted_value[NUMBER_LENGTH_LIMIT];
  unsigned needed_precision = precision;
```

```
  for (int i = 0; i < SIDE_COUNT; i++) {
    if (!numbers_are_equal_by_string(value, sides[i], needed_precision)) {
      continue;
    }

    while (numbers_are_equal_by_string(value, sides[i], needed_precision)) {
      if (needed_precision == MAX_ACTUAL_PRECISION) {
        break;
      }
      needed_precision++;
    }
  }

  unsigned final_precision =
      (needed_precision > precision) ? needed_precision : precision;

  // Extra digit for rounding prevention
  snprintf(formatted_value, sizeof(formatted_value), "%.*Lf",
           final_precision + 1, value);
  truncate_to_precision(formatted_value, final_precision);

  printf("%s %s\n", text, formatted_value);
}
```

utils.c

```
#include "../include/utils.h"
void replace_commas_with_dots(char *str) {
  while (*str) {
    if (*str == ',') {
      *str = '.';
    }
    str++;
  }
}

void flush_input_buffer() {
  int ch;
  while ((ch = getchar()) != '\n' && ch != EOF) {
    // Keep consuming the remaining characters until newline or EOF
  }
}
```

```
void truncate_to_precision(char *formatted_value, unsigned final_precision) {
  for (int i = 0; formatted_value[i] != '\0'; i++) {
    if (formatted_value[i] == '.') {
      // If we reach the decimal point, truncate after the required precision
      if (i + final_precision + 1 < strlen(formatted_value)) {
        formatted_value[i + final_precision + 1] = '\0';
      }
      break;
    }
  }
}
```

constants.h

```
// Side lengths
#define MAX_SIDE_LENGTH (long double)1e6
#define MIN_SIDE_LENGTH (long double)1e-3

// Precision settings
#define MAX_GUARANTEED_PRECISION 15 // Maximum precision for guaranteed
results
#define MAX_ACTUAL_PRECISION 50     // Maximum possible precision
#define MIN_GUARANTEED_PRECISION 1  // Minimum precision

// General tolerance for floating-point comparisons
#define TOLERANCE (long double)1e-15

// Max length of number (whole part + fraction + decimal point)
#define NUMBER_LENGTH_LIMIT (7 + MAX_GUARANTEED_PRECISION + 1)

// Triangle properties
#define SIDE_COUNT 3 // A triangle has 3 sides
#define PROPERTY_COUNT                                          \
  (SIDE_COUNT * 3 + 2) // 3 heights, 3 medians, 3 bisectors, perimeter, area

// Utility macros
#define MAX(a, b) (((a) > (b)) ? (a) : (b))
```

triangle.h

```
#ifndef TRIANGLE_H
#define TRIANGLE_H
```

```c
#include "constants.h"
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// ---------- Function Declarations ----------

// Triangle property calculation functions
long double get_triangle_perimeter(long double a, long double b, long double c);
long double get_triangle_area(long double a, long double b, long double c);
long double get_triangle_height(long double side, long double b, long double c);
long double get_triangle_median(long double side, long double b, long double c);
long double get_triangle_bisector(long double side, long double b,
                                  long double c);

// Input functions
int read_triangle_side(long double *side, const char *side_name);
int read_decimal_places(unsigned *decimal_places);

// Precision handling and printing functions
void print_with_additional_precision(const char *text, unsigned int precision,
                                     long double value,
                                     long double sides[SIDE_COUNT]);

// Triangle validation function
int validate_triangle(long double a, long double b, long double c);

#endif // TRIANGLE_H
```

utils.h

```c
#ifndef UTILS_H
#define UTILS_H

#include "constants.h"
#include <stdio.h>
#include <string.h>

void flush_input_buffer();
void replace_commas_with_dots(char *str);
void truncate_to_precision(char *formatted_value, unsigned final_precision);
```

```
#endif // UTILS_H
```

Makefile

```
# Compiler
CC = clang
CFLAGS = -I. -I$(PATHU) -I$(PATHS) -g -O2 -ffast-math -march=native
LDFLAGS = -lm

# Directories
PATHU = unity/src/
PATHS = src/
PATHT = test/
PATHB = build/
PATHD = build/depends/
PATHO = build/objs/
PATHR = build/results/

# Create all necessary directories
BUILD_PATHS = $(PATHB) $(PATHD) $(PATHO) $(PATHR)

EXECS = triangle
EXECT = triangle_test

# Source files
MAIN_SRCS = $(wildcard $(PATHS)*.c)  # Main source files
TEST_SRCS = $(wildcard $(PATHT)*.c)  # Test source files
UNITY_SRC = $(PATHU)unity.c          # Unity source file

# Object files
MAIN_OBJS = $(MAIN_SRCS:$(PATHS)%.c=$(PATHO)%.o)
TEST_OBJS = $(TEST_SRCS:$(PATHT)%.c=$(PATHO)%.o)
UNITY_OBJ = $(PATHO)unity.o

TARGET_EXTENSION = out

.PHONY: all clean test

all: $(BUILD_PATHS) $(PATHB)$(EXECS).$(TARGET_EXTENSION)

# Build main executable
$(PATHB)$(EXECS).$(TARGET_EXTENSION): $(MAIN_OBJS)
	@echo "Linking main executable..."
```

```
	$(CC) -o $@ $^ $(LDFLAGS)

# Build test executable
$(PATHB)$(EXECT).$(TARGET_EXTENSION): $(TEST_OBJS) $(UNITY_OBJ)
	@echo "Linking test executable..."
	$(CC) -o $@ $^ $(LDFLAGS)

# Compile main source files
$(PATHO)%.o: $(PATHS)%.c
	@echo "Compiling $<..."
	$(CC) $(CFLAGS) -c $< -o $@

# Compile test source files
$(PATHO)%.o: $(PATHT)%.c
	@echo "Compiling $<..."
	$(CC) $(CFLAGS) -DTEST -c $< -o $@  # Define TEST for test compilation

# Compile Unity
$(UNITY_OBJ): $(UNITY_SRC)
	@echo "Compiling Unity..."
	$(CC) $(CFLAGS) -c $< -o $@

# Create directories if they don't exist
$(BUILD_PATHS):
	@mkdir -p $@

# Clean build artifacts
clean:
	@echo "Cleaning up..."
	rm -f $(PATHO)*.o
	rm -f $(PATHB)*.$(TARGET_EXTENSION)
	rm -f $(PATHR)*.txt

# Run tests
test: $(PATHB)$(EXECT).$(TARGET_EXTENSION)
	@echo "Running tests..."
	./$(PATHB)$(EXECT).$(TARGET_EXTENSION)
```

Введені та одержані результати:

Сторони:

1) Showcase

    a) 1e6 1e6 1e-3
2) Цілі + помилка
    a) 13 14 15
    b) 3 4 7
3) Дробові
    a) 3.4 2.3 3.1
    b) 1.3342 1.567 1.826
4) Error cases
    a) 3 4 7 – multiple symbols
    b) -1.2
    c) 228superstring228
    d) 1e7
    e) 1e-4

```
lab2_triangle_props_from_3_sides on  main
› ./build/triangle.out
Вас вітає Простокутник -- інструмент, що допоможе вам знайти:
        - периметр трикутника
        - площу трикутника
        - висоти трикутника
        - медіани трикутника
        - бісектриси трикутника

НАШІ ОБМЕЖЕННЯ:
        - максимальна довжина сторони трикутника: 1e+06
        - максимальна довжина сторони в символах: 23
        - мінімальна довжина сторони трикутника: 0.001
        - максимальна гарантована точність (КІЛЬКІСТЬ знаків після коми) : 15
        - мінімальна гарантована точність: 1
        - застосовується автоматична точність: вона збільшується до тих пір,
                доки довжини не можна буде розрізнити як рядки
        - округлення НЕ застосовується

Для того, щоб вийти з програми передчасно, натисніть Ctrl+C.

Введіть, будь ласка, довжину сторони a: 1e6
Введіть, будь ласка, довжину сторони b: 1e6
Введіть, будь ласка, довжину сторони c: 1e-3

Введіть точність від 1 до 15: 15

========================================
РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:
========================================
Периметр трикутника: 2000000.00099999999997
Площа трикутника: 499.999999988176568
Висота до сторони a: 0.000999999999976
Висота до сторони b: 0.000999999999976
Висота до сторони c: 999999.999976353137697
Медіана до сторони a: 500000.000000000000511
Медіана до сторони b: 500000.000000000000511
Медіана до сторони c: 999999.999999999999886
Бісектриса до сторони a: 0.001414213561295
Бісектриса до сторони b: 0.001414213561295
Бісектриса до сторони c: 999999.999999999999886


PS: Програму можна запускати з деякими параметрами!
-q/--quiet:  вимкнути додатковий вивід у програмі
-r/--repeat: програма повторюватиметься після виконання
-h/--help: вивести цей текст і припинити виконання програми
Дякуємо за використання нашої програми!
Бажаємо Вам гарного дня!

Натисніть Enter для виходу.
```

```
lab2_triangle_props_from_3_sides on  main
› ./build/triangle.out -r -q
Введіть, будь ласка, довжину сторони a: 13
Введіть, будь ласка, довжину сторони b: 14
Введіть, будь ласка, довжину сторони c: 15

Введіть точність від 1 до 15: 15

========================================
РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:
========================================
Периметр трикутника: 42.000000000000000
Площа трикутника: 84.000000000000000
Висота до сторони a: 12.923076923076923
Висота до сторони b: 12.000000000000000
Висота до сторони c: 11.200000000000000
Медіана до сторони a: 12.971121771072847
Медіана до сторони b: 12.165525060596439
Медіана до сторони c: 11.236102527122115
Бісектриса до сторони a: 12.953773111033264
Бісектриса до сторони b: 12.093386622447824
Бісектриса до сторони c: 11.217270634776855

Введіть, будь ласка, довжину сторони a: 3
Введіть, будь ласка, довжину сторони b: 4
Введіть, будь ласка, довжину сторони c: 7

ПОМИЛКА! Сума довжин сторін a та b має бути більшою за довжину сторони c!
Введіть, будь ласка, довжину сторони a: ^C
```

```
Введіть, будь ласка, довжину сторони a: 3.4
Введіть, будь ласка, довжину сторони b: 2.3
Введіть, будь ласка, довжину сторони c: 3.1

Введіть точність від 1 до 15: 15

========================================
РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:
========================================
Периметр трикутника: 8.800000000000000
Площа трикутника: 3.465833233148992
Висота до сторони a: 2.038725431264113
Висота до сторони b: 3.013768028825211
Висота до сторони c: 2.236021440741285
Медіана до сторони a: 2.135415650406262
Медіана до сторони b: 3.043435558706640
Медіана до сторони c: 2.454078238361605
Бісектриса до сторони a: 2.074470861516547
Бісектриса до сторони b: 3.036497515191361
Бісектриса до сторони c: 2.346694367334198

Введіть, будь ласка, довжину сторони a: 1,3342
Введіть, будь ласка, довжину сторони b: 1,567
Введіть, будь ласка, довжину сторони c: 1,826

Введіть точність від 1 до 15: 15

========================================
РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:
========================================
Периметр трикутника: 4.727200000000000
Площа трикутника: 1.020772246612364
Висота до сторони a: 1.530163763472290
Висота до сторони b: 1.302836307099380
Висота до сторони c: 1.118041891141692
Медіана до сторони a: 1.565202891001674
Медіана до сторони b: 1.394026746515288
Медіана до сторони c: 1.133234450588226
Бісектриса до сторони a: 1.555285393975280
```

```
lab2_triangle_props_from_3_sides on  main [!]
› ./build/triangle.out -r -q
Введіть, будь ласка, довжину сторони a: 3 4 7

ПОМИЛКА! Довжина сторони a має бути числом і не містити додаткових символів!
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: -1.2

ПОМИЛКА! Довжина сторони a має бути додатною!
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: 228superstring228

ПОМИЛКА! Довжина сторони a має бути числом і не містити додаткових символів!
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: 1e7

ПОМИЛКА! Довжина сторони a має бути менша-рівна 1e+06.
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: 1e-4

ПОМИЛКА! Довжина сторони a має бути більша-рівна 0.001.
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: 11111111111111111111111111111111111111111111111111111111111111111111111111111

ПОМИЛКА! Довжина сторони a в символах має бути меншою за 23.
Просимо Вас вводити лише програмно коректні сторони трикутника.

Введіть, будь ласка, довжину сторони a: ^C
```

Десяткова частина:

1) 0
2) -12
3) 16
4) 1.3
5) 1e2
6) 2superstring2
7) 6

```
› ./build/triangle.out -q
Введіть, будь ласка, довжину сторони a: 1
Введіть, будь ласка, довжину сторони b: 1
Введіть, будь ласка, довжину сторони c: 1

Введіть точність від 1 до 15: 0

ПОМИЛКА! Точність має бути більша-рівна 1.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: -12

ПОМИЛКА! Точність має бути більша-рівна 1.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 16

ПОМИЛКА! Точність має бути менша-рівна 15.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 1.3

ПОМИЛКА! Точність має бути числом.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 1e2

ПОМИЛКА! Точність має бути числом.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 2superstring2

ПОМИЛКА! Точність має бути числом.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 11111111111111111111111111111111111111111111111111111111111111

ПОМИЛКА! Максимальна довжина значення точності в символах складає 23.
Просимо Вас вводити лише програмно коректну точність.

Введіть точність від 1 до 15: 6

======================================
РЕЗУЛЬТАТИ ОБЧИСЛЕНЬ:
======================================
Периметр трикутника: 3.000000
Площа трикутника: 0.433012
Висота до сторони a: 0.866025
Висота до сторони b: 0.866025
Висота до сторони c: 0.866025
Медіана до сторони a: 0.866025
Медіана до сторони b: 0.866025
Медіана до сторони c: 0.866025
Бісектриса до сторони a: 0.866025
Бісектриса до сторони b: 0.866025
Бісектриса до сторони c: 0.866025
```

Теоретичні розрахунки:

<u>Випадок 1: 1000000 1000000 0.001</u>

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
P	2000000.000999999999976	2000000.0010000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000	0.0000000000689178705215454101562500000000000000000000000000000000000000000000000000000000000000000000
S	499.999999988176569	499.9999999999999999374999999999999999960937499999999999999951171874999999999923706054687499999998664855 96	0.0000000118234310605480670928955078085937499999999999995117187499999999999923706054687499999998664855 96
h_a	0.000999999999999976	0.0009999999999999999999874999999999999999992187499999999999999023437499999999999847412109374999999733	0.0000000000000023646862121096134185791015617187499999999999999023437499999999999847412109374999999733
h_b	0.000999999999999976	0.0009999999999999999999874999999999999999992187499999999999999023437499999999999847412109374999999733	0.0000000000000023646862121096134185791015617187499999999999999023437499999999999847412109374999999733
h_c	999999.999976353137640	999999.9999999999998749999999999999999921874999999999999990234374999999999998474121093749999999732971191406	0.0000236468621210961341857910156171874999999999999990234374999999999998474121093749999999732971191406
m_a	500000.000000000000512	500000.0000000000004999999999999999999975000000000000000024999999999999999687500000000000000043750000000000000	0.0000000000004999999999999999999750000000000000000024999999999999999687500000000000000043750000000000000
m_b	500000.000000000000512	500000.0000000000004999999999999999999975000000000000000024999999999999999687500000000000000043750000000000000	0.0000000000004999999999999999999750000000000000000024999999999999999687500000000000000043750000000000000
m_c	999999.999999999999886	999999.9999999999998749999999999999999921874999999999999990234374999999999998474121093749999999732971191406	-0.0000000000001250000000000000000078125000000000000009765625000000000001525878906250000000267028808594
b_a	0.001414213561296	0.0014142135613124348780383334345586761482409546093868175561927479131279952207816813458904794991502151	0.0000000000000167208751971605496308755245409319119380870874427479131279952207816813458904794991502151
b_b	0.001414213561296	0.0014142135613124348780383334345586761482409546093868175561927479131279952207816813458904794991502151	0.0000000000000167208751971605496308755245409319119380870874427479131279952207816813458904794991502151
b_c	999999.999999999999886	999999.99999999999987499999999999999999921874	-0.00000000000012500000000000000000

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
		99999999999999023437499999999999847412109374 9999999732971191406	7812500000000000000976562500000000 0001525878906250000000267028808594

Випадок 2: 13 14 15

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
P	42.000000000000000	42.000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000 00000000000000	0.000000000000000000000000000000000 0000000000000000000000000000000000 000000000000000000000000000000000
S	84.000000000000000	84.000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000 00000000000000	0.000000000000000000000000000000000 0000000000000000000000000000000000 000000000000000000000000000000000
h_a	12.923076923076923	12.923076923076923076923076923076923076923076 92307692307692307692307692307692307692307692 30769230769231	-0.00000000000000027328566760003853 3027355487530048076923076923076923 0769230769230769230769230769230769
h_b	12.000000000000000	12.000000000000000000000000000000000000000000 00000000000000000000000000000000000000000000 00000000000000	0.000000000000000000000000000000000 0000000000000000000000000000000000 000000000000000000000000000000000
h_c	11.200000000000000	11.200000000000000000000000000000000000000000 00000000000000000000000000000000000000000000 00000000000000	0.000000000000000710542735760100185 8711242675781250000000000000000000 000000000000000000000000000000000
m_a	12.971121771072847	12.971121771072847344753937907671590114902560 67009831374021316593831974653199067032206409 02558603805564	0.000000000000000760137992731915821 1005440401622858137402131659383197 465319906703220640902558603805564
m_b	12.165525060596439	12.165525060596439377999368490404134124169940 18957282237283830609297266545063782047960613 28559156973269	0.000000000000000700137016439097618 1920868103067603223728383060929726 654506378204796061328559156973269
m_c	11.236102527122116	11.236102527122115932299070222745339714980983 08007562819674845683716037832264555979900933 53332561824893	0.000000000000000385922734962752710 4253539078847631281967484568371603 783226455597990093353332561824893
b_a	12.953773111033264	12.953773111033264448163488839546703708759444 29016332143715554266306784536231555941729729 72623322559378	0.000000000000000105668103049302958 8586007528839133214371555426630678 453623155594172972972623322559378
bb	12.093386622447824	12.093386622447824478549919845455656696701594	0.000000000000000781292656725191841

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
		4584128600819048885838958124265385323044402314000758459655	7705541335209128600819048885838958124265385323044402314000758459655
bc	11.217270634776856	11.2172706347768555785931328321304318327818115630740993284325206193734082835782547184143704773908600105	-0.0000000000000004633877308331001403908815185150509006715674793806265917164217452815856295226091399895

Випадок 3: 3.4 2.3 3.1

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
P	8.800000000000000	8.8000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000	-0.000000000000000710542735760100185871124267578125000000000000000000000000000000000000000000000000000
S	3.465833233148993	3.4658332331489927012564082558920281842836939983865203989409738399584716218130910775484730673342403719	0.0000000000000002030271177516506659701952540569802703989409738399584716218130910775484730673342403719
h_a	2.038725431264113	2.0387254312641133536802401505247224613433494108156002346711610823285127187135829867932194513730825717	0.0000000000000001455506110215628985806708945768312252346711610823285127187135829867932194513730825717
h_b	3.013768028825211	3.0137680288252110445707897877321984211162556507708873034269337738769318450548618065638896237689046712	0.0000000000000001572370932688239436188892353870990123034269337738769318450548618065638896237689046712
h_c	2.236021440741286	2.2360214407412856137138117779948568930862541925074325154457895741667558850407039209990148821511228206	-0.0000000000000001125475552392919727573158148992894424845542104258332441149592960790009851178488771794
m_a	2.135415650406262	2.1354156504062622421623047931191421252564575538921160227956212085698017967302816034081296871915567489	-0.000000000000000024248918261307381886168334617328839772043787914301982032697183965918703128084432511
m_b	3.043435558706640	3.0434355587066403335195342156548795649609690833884317196689996879461191254061367858994439471755496756	0.0000000000000001574785424968904626915059214759665567196689996879461191254061367858994439471755496756
m_c	2.454078238361605	2.4540782383616052920374011345539151143468152187332368423007883280511212837422820688560902544306447502	0.0000000000000001899238293598737476777470837734207368423007883280511212837422820688560902544306447502
b_a	2.074470861516548	2.0744708615165475813815645364723247011298024145321276288156265366817555864513311349594402515224291641	-0.0000000000000001114339285282273698131447947534366223711843734633182444135486688650405597484775708359

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
b_b	3.036497515191362	3.0364975151913616557711617850611503845426647017232278734679618631527694416407474249626434851763081098	-0.0000000000000001399011028489883471602228077103861471265320381368472305583592525750373565148236918902
b_c	2.346694367334199	2.3466943673341986325602191977417998700945320323664336444174997905149494115912411585636108599644889893	0.0000000000000001825788330501859047203867378429133086444174997905149494115912411585636108599644889893

Випадок 4: 1.3342 1.567 1.826

	Неточне (з округленням)	Точне (до 100 знаків)	Різниця
P	4.727200000000000	4.7272000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000	0.0000000000000001534772309241816401481628417968750000000000000000000000000000000000000000000000000000
S	1.020772246612365	1.0207722466123647505718966286711809891766949004757023779140086087247806034017598802302502208613775988	-0.000000000000000136458763435167776685648517303836726220859913912752193965982401197697497791386224012
h_a	1.530163763472290	1.5301637634722901372686203397859106418478412538985195291770478319963732624820265031183484048289275953	-0.0000000000000000797694872450267677548539714902421054708229521680036267375179734968816515951710724047
h_b	1.302836307099381	1.3028363070993806644185023977934664826760624128598626393286644655070588428867388388643533648724937	-0.0000000000000000327202106055755817551245357316713873606713355344929411571132611611611356466351275063
h_c	1.118041891141692	1.1180418911416919502430412143167371184848794090642961422935472165660247572856077549071743930573686734	-0.0000000000000001066338361448560578494049063575372663577064527834339752427143922450928256069426313266
m_a	1.565202891001675	1.5652028910016745813051265300076207566650234412124130068090891759429853400029582711607652116733468069	-0.0000000000000000641607421770983622920124210167953994931909108240570146599970417288392347883266531931
m_b	1.394026746515288	1.3940267465152883500466489314783212194324603989087487640143403317728787191753712077044896433767823419	0.0000000000000001102550188554084707448364368393384362640143403317728787191753712077044896433767823419
m_c	1.133234450588227	1.1332344505882266902731334018526859936727600566049690621729724804829833537535620949644792050942558156	0.0000000000000001074497521519433968124403076640268440621729724804829833537535620949644792050942558156

	Неточне (з округленням)	**Точне (до 100 знаків)**	**Різниця**
b_a	1.555285393975281	1.555285393975280549575797114840906636699245354259396061851527980593941806974503719569512088236072589 0	-0.000000000000000010044938648931615747334141931859216643814847201940605819302549628043048791176392741 10
b_b	1.355449754889268	1.355449754889267754979026853192526018982127813069997311931003151187124126486539042952235648547561065 1	0.000000000000000088820978766392844832306102422444997311931003151187124126486539042952235648547561065 1
b_c	1.123605648909320	1.123605648909320113456924638698149105928949930504893950144674790429551867454099845248021917254672009 1	0.000000000000000092878263068954731902502588846520518950144674790429551867454099845248021917254672009 1

Як бачимо, значення відрізняються не більш ніж на 10^{-15}(крім першого випадку, де похибка відбувається через ділення чисел, що сильно відрізняються порядком), що, очевидно, забезпечується типом *long double*.

Висновки:

Теоретичні розрахунки відповідають отримані з точністю не менше за 15 знаків після коми. Програма працює коректно, а користувач отримує теплий та приємний UX, що враховує знакові уподобання (кома чи крапка) та автоматично збільшує кількість знаків після коми в разі їх співпадіння з сторонами трикутника.