ФГОБУ ВПО "СибГУТИ"
Кафедра вычислительных систем

ПРОГРАММИРОВАНИЕ
ЯЗЫКИ ПРОГРАММИРОВАНИЯ

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ
"Последовательные программы"

Преподаватель:
Доцент Кафедры ВС, к.т.н.
Поляков Артем Юрьевич

Структура простейшей программы на языке СИ

hello.c

```
#include <stdio.h>

// Коментарий
int main()
{
    // Вывод сообщения в кавычках на экран
    printf("Hello world\n");

    // Возврат нулевого кода – "успешное завершение"
    return 0;
}
```

Командная строка:

```
$ gcc –o hello hello.c
```

Вычисление суммы двух целых чисел

summ.c

```
#include <stdio.h>

int main()
{
    int a, b;
    a = 5;
    b = 10;
    printf("%d\n", a + b );

    return 0;
}
```

Командная строка:

```
$ gcc –o summ summ.c
```

Вычисление суммы двух целых чисел

summ1.c

```
#include <stdio.h>

int main()
{
    int a = 5, b = 10, c;

    c = a + b;
    printf("a + b = %d\n", c );

    return 0;
}
```

Командная строка:

```
$ gcc –o summ_1 summ1.c
```

Вычисление суммы двух целых чисел

summ_input.c

```
#include <stdio.h>

int main()
{
    int a, b;
    printf("Input a, b\n");
    scanf("%d %d",&a, &b);
    printf("%d + %d = %d\n",a, b, a+b);

    return 0;
}
```

Командная строка:

```
$ gcc –o summ_in summ_input.c
```

Вычисление дискриминанта квадратного уравнения

$$a \cdot x^2 + b \cdot x + c = 0$$

$$D = b^2 - 4 \cdot a \cdot c$$

Вопросы:

1. Какие типы данных использовать?
2. Что является входными данными для данной задачи?

Вычисление дискриминанта квадратного уравнения

discrim.c

```
#include <stdio.h>

int main()
{
    float a, b, c, D;
    printf("Input a, b, c\n");
    scanf("%f %f %f", &a, &b, &c);

    D = b*b - 4*a*c;

    printf("D = %f\n", D);

    return 0;
}
```

В качестве разделителя между целой и дробной частями выступает **точка:**

1.2

3.0

125.6544

Командная строка:

```
$ gcc -o discrim discrim.c
```

Примеры входных данных

1)

$$x^2 + 2 \cdot x + 1 = 0$$

$$D = 2^2 - 4 \cdot 1 \cdot 1 = 4 - 4 = 0$$

2)

$$2 \cdot x^2 + 5 \cdot x + 3 = 0$$

$$D = 5^2 - 4 \cdot 2 \cdot 3 = 25 - 24 = 1$$

3)

$$8 \cdot x^2 + 3 \cdot x - 4 = 0$$

$$D = 3^2 + 4 \cdot 8 \cdot 4 = 9 + 128 = 137$$

4)

$$8 \cdot x^2 + 3 \cdot x + 4 = 0$$

$$D = 3^2 - 4 \cdot 8 \cdot 4 = -119$$

Задание

1. В строке ответа вывести формулу, по которой вычислен результат, оставив в ней переменные (файл discrim_fout1.c)

2. В строке ответа вывести формулу, по которой вычислен результат с подстановкой введенных значений (файл discrim_fout2.c)

Программный отладчик GNU GDB

http://www.gnu.org/software/gdb/
http://ru.wikipedia.org/wiki/GNU_Debugger

Переносимый отладчик проекта GNU, который работает на многих UNIX-подобных системах и умеет производить отладку многих языков программирования, включая:

- Си;
- C++;
- Free Pascal;
- FreeBASIC;
- Ada;
- Фортран.

GDB — свободное программное обеспечение, распространяемое по лицензии GPL.

Работа с GDB

1. При компиляции программы необходимо указать специальный ключ: **-g**

```
$ gcc –g –o discrim discrim.c
```

2. Запуск программы производится под контролем GDB:

```
$ gdb discrim
GNU gdb (GDB) 7.2-ubuntu
Copyright (C) 2010 Free Software Foundation, Inc.
. . . . . .
.../discrim...done.
(gdb)
```

Работа с GDB

3. Листинг кода программы (команда **`list`**, сокр. **`"l"`**)

```
(gdb) list
1       #include <stdio.h>
2
3       int main()
4       {
5           float a, b, c, D;
6           printf("Input a, b, c\n");
7           scanf("%f %f %f", &a, &b, &c);
8
9           D = b*b - 4*a*c;
10
```

Работа с GDB

4. Запуск программы в GDB без трассировки

```
(gdb) r
Starting program: .../discrim
Input a, b, c
1 2 1
D = 0.000000

Program exited normally.
(gdb)
```

Данный режим не позволяет выполнять никаких отладочных действий, если программа выполняется без критичных ошибок (приводящих к аварийному завершению программы)

Трассировка программы

Трассировка — процесс пошагового выполнения программы.

В режиме трассировки программист **видит последовательность выполнения команд** и **имеет доступ к значениям переменных** на данном шаге выполнения программы, что позволяет легче обнаруживать ошибки.

Трассировка может быть начата и окончена в любом месте программы, выполнение программы может останавливаться на **каждой команде** или на **точках останова**.

Трассировка может выполняться с **заходом в процедуры и без заходов**.

Вычисление дискриминанта квадратного уравнения

discrim.c

точка 1 

точка 2 

```
#include <stdio.h>

int main()
{
    float a, b, c, D;
    printf("Input a, b, c\n");
    scanf("%f %f %f", &a, &b, &c);

    D = b*b - 4*a*c;

    printf("D = %f\n", D);

    return 0;
}
```

Работа с GDB

5. Установка точки останова (команда **break**, сокр. **"b"**):
1) на указанной **функции**; 2) на указанной **строке**.

```
(gdb) l
1       #include <stdio.h>
2
3       int main()
4       {
5           float a, b, c, D;
6           printf("Input a, b, c\n");
7           scanf("%f %f %f", &a, &b, &c);
8
9           D = b*b - 4*a*c;
10
(gdb) b main
Breakpoint 1 at 0x804844d: file discrim.c, line 6.
(gdb) b 9
Breakpoint 2 at 0x804847e: file discrim.c, line 9.
```

Работа с GDB

6. Запуск программы с трассировкой по точкам останова. Продолжение выполнения (команда **`continue`** сокр. "**`c`**")

```
(gdb) r
Starting program: .../discrim
Breakpoint 1, main () at discrim.c:6
6           printf("Input a, b, c\n");
(gdb) continue
Continuing.
Input a, b, c
1 2 1
Breakpoint 2, main () at discrim.c:9
9           D = b*b - 4*a*c;
(gdb) c
Continuing.
D = 0.000000

Program exited normally.
```

Работа с GDB

7. Получение доступа к значениям переменных (команда **inspect**)

```
(gdb) r
Starting program: .../discrim
Breakpoint 1, main () at discrim.c:6
6           printf("Input a, b, c\n");
(gdb) continue
Continuing.
Input a, b, c
1 2 1
Breakpoint 2, main () at discrim.c:9
9           D = b*b - 4*a*c;
(gdb) inspect a
$1 = 1
(gdb) inspect b
$2 = 2
(gdb) inspect c
$3 = 1
```

Работа с GDB

8. Просмотр и удаление точек останова:
команды **`info breakpoints`** и **`delete`**

```
(gdb) info breakpoints
Num Type           Disp Enb Address    What
1   breakpoint keep y   0x0804844d in main at discrim.c:6
2   breakpoint keep y   0x0804847e in main at discrim.c:9

(gdb) delete 2

(gdb) info breakpoints
Num Type           Disp Enb Address    What
1   breakpoint keep y   0x0804844d in main at discrim.c:6
```

Вычисление дискриминанта квадратного уравнения

discrim.c

точка 1

точка 2

```
#include <stdio.h>

int main()
{
    float a, b, c, D;
    printf("Input a, b, c\n");
    scanf("%f %f %f", &a, &b, &c);

    D = b*b - 4*a*c;

    printf("D = %f\n", D);

    return 0;
}
```

Работа с GDB

9. Пошаговая трассировка программы с остановкой на каждом выражении (команда **next** сокр. "**n**")

```
(gdb) r
. . .
(gdb) n
Input a, b, c
7           scanf("%f %f %f", &a, &b, &c);
(gdb) n
1 2 1
9           D = b*b - 4*a*c;
(gdb) inspect b
$1 = 2
(gdb) n
11          printf("D = %f\n", D);
(gdb) inspect D
$2 = 0
(gdb) n
D = 0.000000
```

Вопрос?

Чего не хватает для полноценного решения квадратного уравнения?