Практическое занятие 4
Управляющие конструкциии

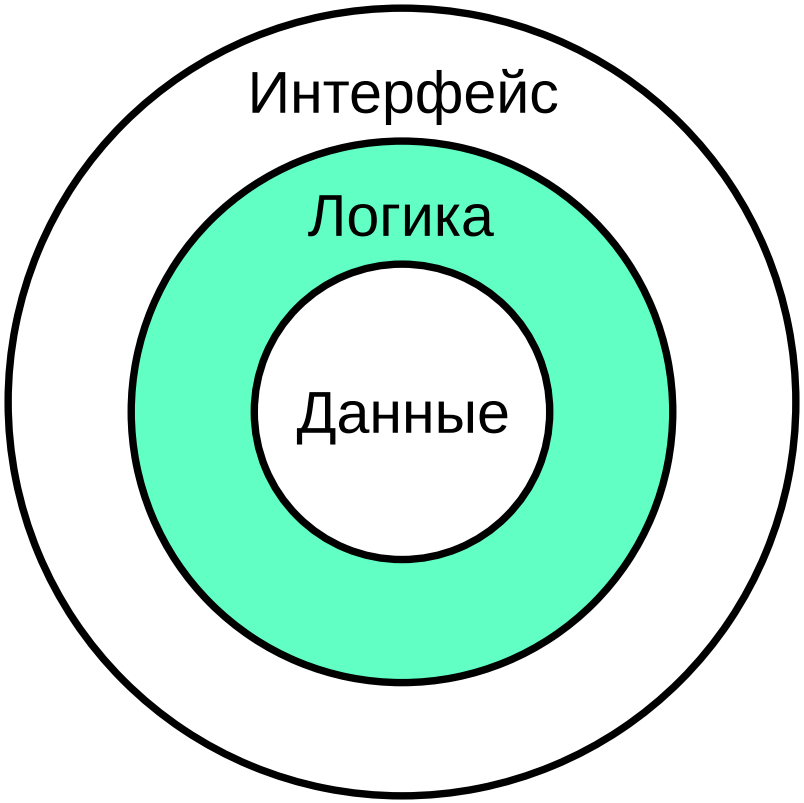
Пименов Евгений Сергеевич

Курс «Программирование»

Сибирский государственный университет телекоммуникаций и информатики (Новосибирск)

Осенний семестр, 2016

Архитектура абстрактного приложения



Управляющие конструкции

- `if-else`
- `else-if`
- `while, for`
- `do-while`
- `break, continue`
- ~~`goto`~~

Оператор `if-else`


```
01. if (выражение) {
02.     // Если выражение принимает значение true
03.     оператор
04. } else {
05.     // Если выражение принимает значение false
06.     оператор
07. }
```

Конструкция else-if


```
01. if (выражение 1) {
02.     оператор
03. } else if (выражение 2) {
04.     оператор
05. } else {
06.     оператор
07. }
```

Сравните


```
01. if (выражение 1) {
02.     оператор
03. } else if (выражение 2) {
04.     оператор
05. }
```

```
01. if (выражение 1) {
02.     оператор
03. }
04. if (выражение 2) {
05.     оператор
06. }
```

Логические операторы

- Операторы отношения: `<`, `<=`, `>`, `>=`, `==`, `!=`
- Логические операторы: `!`, `||`, `&&`

Логические И (&&), ИЛИ (||)

A	B	A && B	A || B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Laziness


```
01. #include <stdio.h>
02.
03. int f()
04. {
05.     printf("f()\n");
06.     return 1;
07. }
08.
09. int g()
10. {
11.     printf("g()\n");
12.     return 0;
13. }
14.
15. int main()
16. {
17.     if (f() || g()) {  }
18.
19.     if (g() && f()) {  }
20.
21.     return 0;
22. }
```

A	B	A && B	A || B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Логический тип

Стандарт языка	Тип
C89	int
C99	_Bool

Тип `_Bool` доступен под именем `bool` при подключении заголовочного файла `stdbool.h`

Логический тип

- Ложь – 0
- Истина – любое ненулевое значение

```
01. if (42 && -10e5 && 'Z') {
02.     printf("true\n");
03. }
```

Логический тип


```
01. #include <stdio.h>
02.
03. int main()
04. {
05.     int true_val, false_val;
06.
07.     true_val = 42 > 10;
08.     false_val = !42;
09.
10.     printf("42 > 10 = %d\n", true_val);   // 42 > 10 = 1
11.     printf("!42 = %d\n", false_val);      // !42 = 0
12.
13.     return 0;
14. }
```

Результат работы фрагмента кода?


```
01. int x = 0;
02.
03. if (x = 42) {
04.     printf("x = 42");
05. }
```

Цикл while


```
01. while (<условие продолжения>) {
02.     [операторы]
03. }
```

Цикл while


```
01. #include <stdio.h>
02.
03. int main()
04. {
05.     int num, sum = 0, status;
06.
07.     printf("Type integer number or q to exit\n");
08.
09.     status = scanf("%d", &num);
10.
11.     while (status == 1) {
12.         sum += num;
13.         status = scanf("%d", &num);
14.     }
15.
16.     printf("Sum: %d\n", sum);
17.
18.     return 0;
19. }
```

Цикл while


```
01. #include <stdio.h>
02.
03. int main()
04. {
05.     int num, sum = 0;
06.
07.     printf("Type integer number or q to exit\n");
08.
09.     while (scanf("%d", &num) == 1) {
10.         sum += num;
11.     }
12.
13.     printf("Sum: %d\n", sum);
14.
15.     return 0;
16. }
```

Цикл while


```
01.     fahr = lower;
02.
03.     while (fahr <= upper) {
04.         celsius = 5 * (fahr - 32) / 9;
05.         printf("%d\t%d\n", fahr, celsius);
06.         fahr += step;
07.     }
```

Цикл while


```
01.     fahr = lower;
02.
03.     while (fahr <= upper) {
04.         celsius = 5 * (fahr - 32) / 9;
05.         printf("%d\t%d\n", fahr, celsius);
06.         fahr += step;
07.     }
```

Цикл for


```
    fahr = lower;

    while (fahr <= upper) {
        celsius = 5 * (fahr - 32) / 9;
        printf("%d\t%d\n", fahr, celsius);
        fahr += step;
    }
```

```
    for (fahr = lower; fahr <= upper; fahr += step) {
        celsius = 5 * (fahr - 32) / 9;
        printf("%d\t%d\n", fahr, celsius);
    }
```

Цикл for


```
01.     fahr = lower;
02.
03.     while (fahr <= upper) {
04.         celsius = 5 * (fahr - 32) / 9;
05.         printf("%d\t%d\n", fahr, celsius);
06.         fahr += step;
07.     }
```

```
01.     for (fahr = lower; fahr <= upper; fahr += step) {
02.         celsius = 5 * (fahr - 32) / 9;
03.         printf("%d\t%d\n", fahr, celsius);
04.     }
```

Цикл for


```
01.     fahr = lower;
02.
03.     while (fahr <= upper) {
04.         celsius = 5 * (fahr - 32) / 9;
05.         printf("%d\t%d\n", fahr, celsius);
06.         fahr += step;
07.     }
```

```
01.     for (fahr = lower; fahr <= upper; fahr += step) {
02.         celsius = 5 * (fahr - 32) / 9;
03.         printf("%d\t%d\n", fahr, celsius);
04.     }
```

Цикл for


```
01.     fahr = lower;
02.
03.     while (fahr <= upper) {
04.         celsius = 5 * (fahr - 32) / 9;
05.         printf("%d\t%d\n", fahr, celsius);
06.         fahr += step;
07.     }
```

```
01.     for (fahr = lower; fahr <= upper; fahr += step) {
02.         celsius = 5 * (fahr - 32) / 9;
03.         printf("%d\t%d\n", fahr, celsius);
04.     }
```

Цикл `for`


```
01. for (init expression; cond_expression; iter_expression) {
02.     оператор
03. }
```

Цикл for


```
01. int i;
02.
03. for (i = 0; i < 5; ++i) {
04.     printf("%d ", i);
05. }
```

Вывод:

```
0 1 2 3 4
```

Цикл do-while


```
01. do {
02.     [операторы]
03. } while (<условие продолжения>);
```

Сколько раз будет выполнено тело цикла? 26

```
01. while (1) {
02.     printf("Step...\n");
03. }
```

```
01. do {
02.     printf("Step...\n");
03. } while (0);
```

Сколько раз будет выполнено тело цикла?

```
01. while (0) {
02.     printf("Step...\n");
03. }
```

```
01. for (;;) {
02.     printf("Step...\n");
03. }
```

Операторы break, continue в циклах

- `break` — выход из цикла
- `continue` — переход на следующую итерацию

Задание 1

Разработать приложение для рисования лестницы. Пользователь вводит количество ступенек. Использовать цикл `for`.

```
$ ./steps
Steps: 4
    #
   ##
  ###
 ####
```

Задание 2

Разработать приложение для поиска минимума и максимума во входной последовательности. Приложение в интерарактивном режиме запрашивает у пользователя числа. Завершение ввода — любая нечисловая последовательность (например, символ q). После завершения ввода приложение выводит максимальный и минимальный элементы и их номера.

Hint: для упрощения тестирования входные данные удобно хранить в файле. Google: bash io redirection.

Домашнее задание 1

Разработать приложение для вычисления квадратного корня методом Ньютона по формуле:

$$x_{i+1} = \frac{x_i + \frac{n}{x_i}}{2}$$

Где:

- n — число, из которого извлекается корень
- x_i — предыдущее значение корня
- x_{i+1} — уточненное значение корня

Вычисления производить с точностью 10^{-4}.

Домашнее задание 2

Разработать программу для вывода пирамиды из букв в верхнем регистре.

Реализовать проверку входных данных. См. `man isalpha`. Пример использования:

```
$ ./pyramid
Lower letter: E
    A
   ABA
  ABCBA
 ABCDCBA
ABCDEDCBA
```

Домашнее задание 3*

Решить, используя арифметические, логические операции и операции сравнения. Разработать программы, обеспечивающие проверку попадания заданной координаты в указанные интервалы:

1. $x \in [0; +\infty)$
2. $x \in [5; 15)$
3. $x \in (-1; 1)$
4. $x \in (-1; 1) \cup [5; 15)$
5. $x \in (-1; 1) \cup [5; 15) \cup \{20, 100, 1000\}$

Без использования оператора ветвления