

Лекция 5

Примеры решения задач

Пример 1: Представление целого числа в двоичном виде в виде строки

[illegible]
$$255_{10} \rightarrow 0000000000000000000000000000000011111111_2$$

- См. функции

```
public static String intToBinV0(int n)
```

```
public static String intToBin(int n)
```

в проекте Lect5Samples

Представление целого числа в двоичном виде (в виде строки)

```
/* Собственная реализация представления int
 * в бинарном виде в виде строки
 */
public static String intToBinV0(int n) {
    String result = "";
    for (int i = Integer.SIZE - 1; i >= 0; i--)
        result += (n >> i) & 1;
    return result;
}
```

- Integer.*SIZE* – размер целого числа в битах (= 32)
- Цикл *for* пробегает от 31 до 0 (последовательные значения переменной *i*)
- Для примера возьмем $n = 11_{10}$ ($00...001011_2$) и $i = 1$:
 - $00...0010\mathbf{1}_2 \gg 1 = 00...00010\mathbf{1}_2$
 - $00...00010\mathbf{1}_2 \& 00...00000\mathbf{1}_2 = 00...00000\mathbf{1}_2 = \mathbf{1}_{10}$

Представление целого числа в двоичном виде (с доработками)

```
/* Собственная реализация представления int
 * в бинарном виде в виде строки
 * (при множественной конкатенации строк
 * эффективнее использовать StringBuilder)
 */
public static String intToBin(int n) {
    StringBuilder result = new StringBuilder();
    for (int i = Integer.SIZE - 1; i >= 0; i--)
        result.append((n >> i) & 1);
    return result.toString();
}
```

- Строки (String) в Java – неизменяемые объекты (после создания содержимое строки не может быть ни каким образом изменено)
- Любая конкатенация строк приводит к созданию нового объекта строки (в примере `intToBinV0` будет создано 33 строки)
- Создание объектов – «тяжелая» операция, множественного повторения которой следует избегать

Пример 2: собственная реализация sqrt (метод половинного деления)

- См. функцию

```
public static double sqrt(double x)
```

в проекте Lect5Samples

```

final double EPS = 1E-9;
...
public static double sqrt(double x) {
    if (x < 0) {
        return Double.NaN;
    }

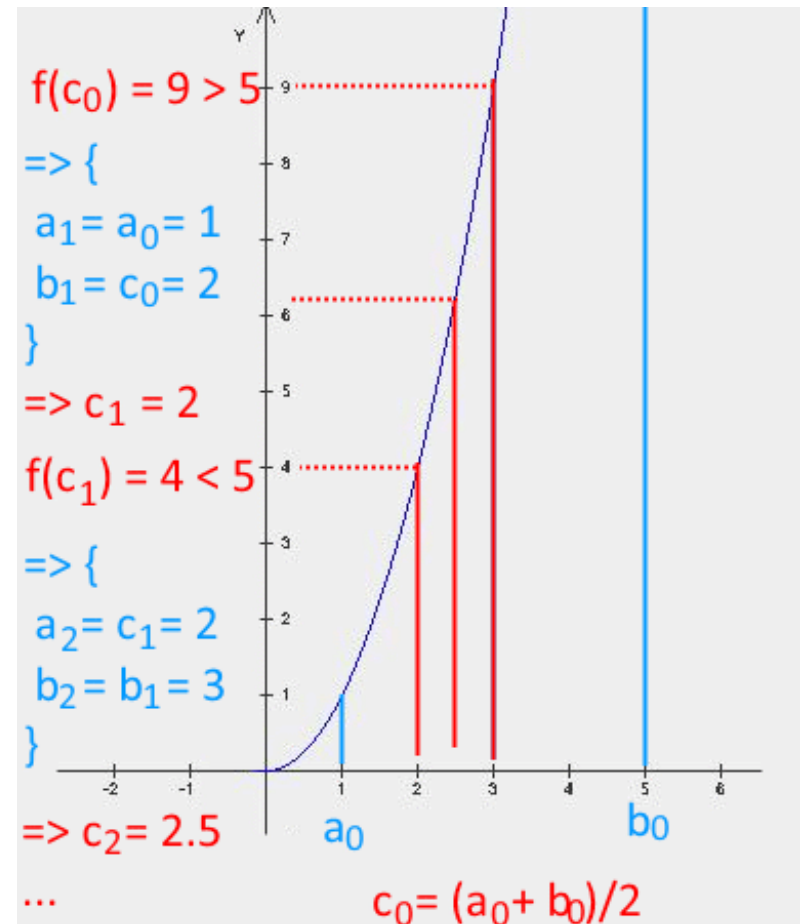
    // double a = (x < 1) ? 0 : 1,
    //           b = (x < 1) ? 1 : x;
    double a = 1, b = x;
    if (x < 1) {
        a = 0;
        b = 1;
    }

    while (b - a > EPS) {
        double c = (a + b) / 2;
        if (c * c > x) {
            b = c;
        } else {
            a = c;
        }
    }

    return (a + b) / 2;
}

```

- Надо найти $y = \sqrt{x}$
- Если $x < 0$, то решения нет (Double.NaN)
- Если $x \geq 1$, то $1 \leq y < x$
- Если $0 \leq x < 1$, то $0 \leq y < 1$
- Рассмотрим обратную функцию $y' = f(x') = x'^2$
- Надо подобрать такой x' , чтобы y' был равен x (из $y = \sqrt{x}$)



Пример 3: Вычисление числа Пи

- Плохо сходятся

- Формула Валлиса:

$$\frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \cdot \frac{8}{7} \cdot \frac{8}{9} \cdots = \frac{\pi}{2}$$

- Ряд Лейбница:

$$\frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots = \frac{\pi}{4}$$

- Хорошо сходится

- Формула Виета для приближения числа π

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2} \cdot \frac{\sqrt{2 + \sqrt{2}}}{2} \cdot \frac{\sqrt{2 + \sqrt{2 + \sqrt{2}}}}{2} \cdots$$

- См. функции

```
public static double pi()
```

```
public static double pi(int n)
```

Пример 3: Вычисление числа Пи

- Формула Виета для приближения числа π

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2} \cdot \frac{\sqrt{2 + \sqrt{2}}}{2} \cdot \frac{\sqrt{2 + \sqrt{2 + \sqrt{2}}}}{2} \cdot \dots$$

- Произведение в правой части сходится к $2/\pi$, очевидно, что чтобы такое происходило правые множители должны сходиться к 1

- Обозначим на шаге $(i - 1)$

– num_{i-1} и mult_{i-1}

- А на шаге i

– num_i и mult_i

- Очевидно, что:

- $\text{mult}_i = \text{mult}_{i-1} * \text{num}_{i-1} / 2$
- $\text{num}_i = \text{Math.sqrt}(2 + \text{num}_{i-1})$

```
public static double pi(int n)
{
    double mult = 1;
    double num = Math.sqrt(2);
    for (int i = 0; i < n; i++) {
        mult = mult * num / 2;
        num = Math.sqrt(2 + num);
    }

    return 2 / mult;
}
```


Пример 3: Вычисление числа Пи

- Формула Виета для приближения числа π

$$\frac{2}{\pi} = \frac{\sqrt{2}}{2} \cdot \frac{\sqrt{2 + \sqrt{2}}}{2} \cdot \frac{\sqrt{2 + \sqrt{2 + \sqrt{2}}}}{2} \cdot \dots$$

- Учитывая, что мы не знаем, какое кол-во шагов n необходимо сделать для приемлемой точности (очевидно, что чем n больше, тем точнее), можно задать кол-во шагов через EPS, где EPS будет задавать отличие `num` от 2 (очевидно, что `num` должно сходиться к числу 2)

```
/* вычисление числа Пи
 * по формуле Виета
 */
public static double pi()
{
    double mult = 1;
    double num = Math.sqrt(2);
    for (Math.abs(num - 2) > EPS)
    {
        mult = mult * num / 2;
        num = Math.sqrt(2 + num);
    }

    return 2 / mult;
}
```

Пример 4: Выделение N-ой части строки, разделенной запятыми

```
getStringPart(" 1, , abc , ", 0) -> "1"  
getStringPart(" 1, , abc , ", 1) -> ""  
getStringPart(" 1, , abc , ", 2) -> "abc"  
getStringPart(" 1, , abc , ", 3) -> ""  
getStringPart(" 1, , abc , ", 4) -> null  
getStringPart(" 1, , abc , ", -1) -> null
```

- См. функцию

```
public static String getStringPart(String str,  
                                     int partIndex)
```

в проекте Lect5Samples

Пример 4: Выделение N-ой части строки, разделенной запятыми

```
public static String getStringPart(String str, int partIndex) {
    int lastCommaPos = -1;
    for (int i = 0; i < partIndex; i++) {
        int commaPos = str.indexOf(",", lastCommaPos + 1);
        if (commaPos < 0) {
            return null;
        }
        lastCommaPos = commaPos;
    }
    int commaPos = str.indexOf(",", lastCommaPos + 1);
    if (commaPos < 0) {
        return str.substring(lastCommaPos + 1).trim();
    } else {
        return str.substring(lastCommaPos + 1, commaPos).trim();
    }
}
```

Пример 5

- Найти N-ое по счету число, такое сумма всех цифр не превышает S

(Строки и массивы не использовать)

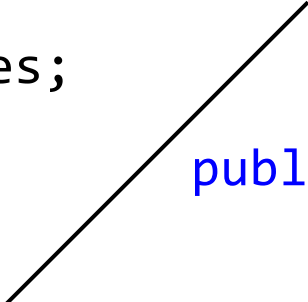
- См. функцию

```
public static int sample5(int n, int s)
```

в проекте Lect5Samples

Найти N-ое по счету число среди чисел,
сумма всех цифр которых не превышает S

```
static int digitsSum(int n, int s) {  
    int res = 0;  
    while (n > 0)  
    {  
        res += n % 10;  
        n /= 10;  
    }  
    return res;  
}
```



```
public static int sample5(int n, int s) {  
    int index = 0;  
    for (int i = 0; i <= Integer.MAX_VALUE; i++) {  
        if (digitsSum(i) <= s) {  
            index++;  
            if (index == n)  
                return i;  
        }  
    }  
    return -1;  
}
```

Пример 6: Печать N символов последовательности

abcdaabbccddabcdaabbccddabcdaabbcc...

- См. функцию

```
public static void sample6_1(int x)
```

abcdaabbccddaaabbbccdddaaaabbbbcccd...

- См. функцию

```
public static void sample6_2(int x)
```

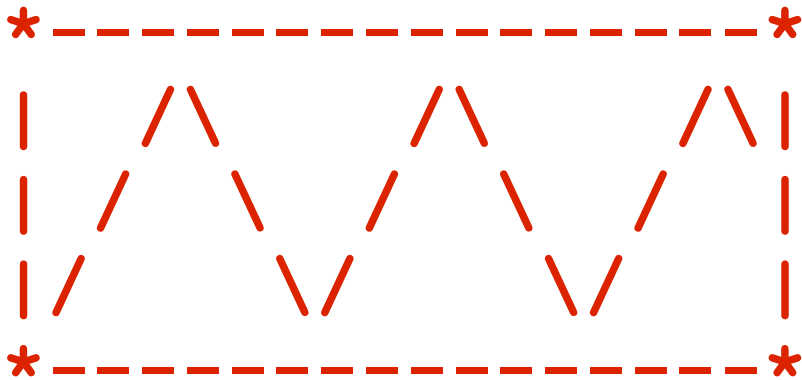
abbccddddeeeeeeffffffggggggghhhhhhhhhi...

- См. функцию

```
public static void sample6_3(int x)
```

Пример 7. «Нарисовать»

$w \geq 3, h \geq 3$ ($w = 18, h = 5$)



- См. функцию

```
public static void sample7(int w, int h)
```

в проекте Lect5Samples

Пример 8. Распечатать

- $h \geq 1$ ($h = 16$)

abcaabbccaaabbbbc
ccaaaabbbbcccca
aaaabbbbcccccc
aaaaabbbbbbbc
cccccaaaaaa
bbbbbbbcccc
cccaaaaaa
abbbbbbbb
cccccccc
aaaaaaa
aabbbb
bbbb
cccc
ccc
cc
a

- См. функции

```
public static void sample8(int h)
public static void sample8_v2(int h)
```

в проекте Lect5Samples

Совет (настоятельная рекомендация)

- Обязательно смотрите проекты-примеры к лекции (и экспериментируйте с ними):
 - Lect5Samples