Современные проблемы информатики

Задача №1

Кодирование целых чисел.

МГ-101 Тимофеев Д.А.

СОДЕРЖАНИЕ:

1. Задание

Запрограммировать коды $\varphi_0, \varphi_1, \varphi_2$, с соответствующими декодерами. Проверить корректность их работы на тестовых последовательностях целых чисел.

2. Программный код

2.1. Вспомогательный код

Для кодирования порций бит, некратных байту был создан вспомогательный класс ChunkBits.

```
public interface ChunkBits {
    public int size();

    public boolean getBitFromIndex(int index);

    public byte[] getAllContent();

    public ChunkBits getBitsFromInfiniteArray(int fromBit, int toBit) throws Exception;

    public String getBinareString();

    public ChunkBits concatinateWith(ChunkBits chunkBitsConcatinate) throws Exception;

    public void addInEnd(ChunkBits chunkBits) throws Exception;

    public void addOneInStart() throws Exception;

    public void deleteZerosInStart() throws Exception;

    public Integer getPositionFirstOne();

    public int getInt() throws Exception;

    public boolean isEmpty() throws Exception;
```

2.2. Fi0 кодер

```
public ChunkBits code(ChunkBits number) throws Exception {
    if (number.equals(new ChunkBitsClass("0"))) {
        return new ChunkBitsClass("1");
    }

    return f0(number);
}
private ChunkBits f0(ChunkBits number) throws Exception {
    int lengthNumber = number.getInt();
    assert(lengthNumber >= 0);

    String codeNumber = "";
    for (int i = 0; i < lengthNumber; i++) {
        codeNumber += "0";
    }

    codeNumber += "1";
    return new ChunkBitsClass(codeNumber);
}
```

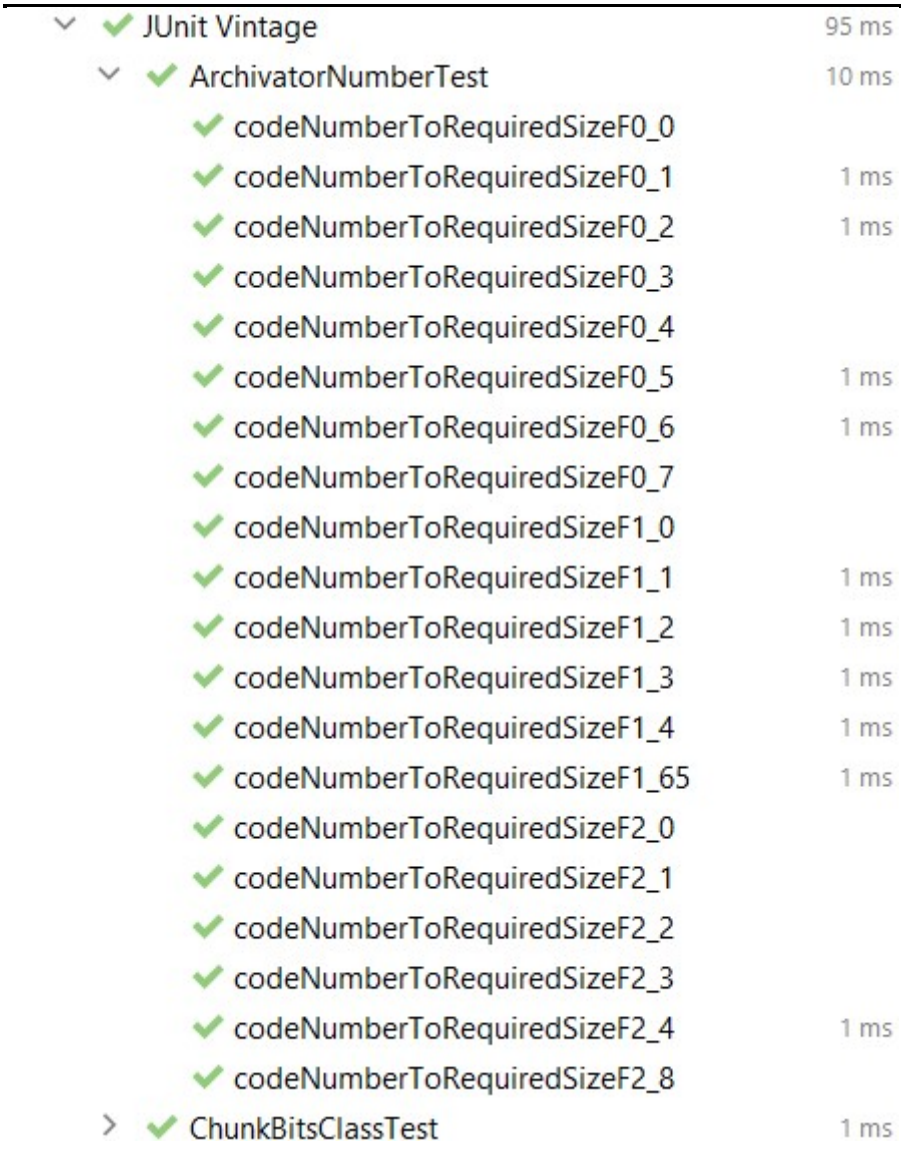
2.3. Fi > 0 кодер

```
private ChunkBits codeFiMore0(int fi, ChunkBits number) throws Exception {
    if (fi == 0) {
        return this.fi0.code(number);
    } else if (number.equals(new ChunkBitsClass("0"))) {
        return new ChunkBitsClass("1");
    } else if (number.equals(new ChunkBitsClass("1"))) {
        return new ChunkBitsClass("01");
    } else {
        ChunkBits codeNumberBitsForRead = codeFiMore0(fi - 1, new
ChunkBitsClass(number.size()));
        ChunkBits withoutFirtOne = number.getBitsFromInfiniteArray(1, number.size());
        return codeNumberBitsForRead.concatinateWith(withoutFirtOne);
    }
}
```

2.4. Fi декодер

```
private ChunkBits decodeFi(int fi) throws Exception {

    ChunkBits res = new ChunkBitsClass(null, 0);
    Integer startReadNextData = fi0Decode();
    if (startReadNextData == null) {
        return res;
    }

    if (fi == 0 || (startReadNextData < 2)) {
        this.endFirstCodeNumber = startReadNextData;
        return new ChunkBitsClass(startReadNextData);
    }

    int sizeNextChunk = startReadNextData;
    int endReadNextData = startReadNextData + sizeNextChunk;
    startReadNextData++; //потому что первая единица относиться к f0 //FIXME

    for (int i = 0; i < fi; i++) {
        res = buffer.getBitsFromInfiniteArray(startReadNextData, endReadNextData);
        int impliedOneInStartInNextChunk = 1;
        res.addOneInStart();
        ChunkBits chunkWithImpliedOneInStart = res;
        sizeNextChunk = chunkWithImpliedOneInStart.getInt() -
impliedOneInStartInNextChunk;
        startReadNextData = endReadNextData;
        endReadNextData = startReadNextData + sizeNextChunk;
    }


    this.endFirstCodeNumber = startReadNextData;


    return res;
}
```

3. Тесты




```
@Test
public void codeNumberToRequiredSizeF0_0() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "0", "1"));
}

@Test
public void codeNumberToRequiredSizeF0_1() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "1", "01"));
}

@Test
public void codeNumberToRequiredSizeF0_2() throws Exception {
```

```
    assert (testCodeNumberToRequiredSize(0, "10", "001"));
}

@Test
public void codeNumberToRequiredSizeF0_3() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "11", "0001"));
}

@Test
public void codeNumberToRequiredSizeF0_4() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "100", "00001"));
}

@Test
public void codeNumberToRequiredSizeF0_5() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "101", "000001"));
}

@Test
public void codeNumberToRequiredSizeF0_6() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "110", "0000001"));
}

@Test
public void codeNumberToRequiredSizeF0_7() throws Exception {
    assert (testCodeNumberToRequiredSize(0, "111", "00000001"));
}

@Test
public void codeNumberToRequiredSizeF1_0() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "0", "1"));
}

@Test
public void codeNumberToRequiredSizeF1_1() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "1", "01 "));
}

@Test
public void codeNumberToRequiredSizeF1_2() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "10", "001 0"));
}

@Test
public void codeNumberToRequiredSizeF1_3() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "11", "001 1"));
}

@Test
public void codeNumberToRequiredSizeF1_4() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "100", "0001 00"));
}
@Test
```

```
public void codeNumberToRequiredSizeF1_65() throws Exception {
    assert (testCodeNumberToRequiredSize(1, "1000001", "00000001
000001"));
}

@Test
public void codeNumberToRequiredSizeF2_0() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "0", "1"));
}

@Test
public void codeNumberToRequiredSizeF2_1() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "1", "01"));
}

@Test
public void codeNumberToRequiredSizeF2_2() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "10", "001 0 0"));
}

@Test
public void codeNumberToRequiredSizeF2_3() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "11", "001 0 1"));
}

@Test
public void codeNumberToRequiredSizeF2_4() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "100", "001 1 00"));
}

@Test
public void codeNumberToRequiredSizeF2_8() throws Exception {
    assert (testCodeNumberToRequiredSize(2, "1000", "0001 00 000"));
}
```

4. Ссылка на исходники

SibGUTY_git/5k1s/СПИ - Современные проблемы информатики (Фионов)/labs_realisation/lab_1-3 at master · GeorgiaFrankinStain/SibGUTY_git
https://github.com/GeorgiaFrankinStain/SibGUTY_git/tree/master/5k1s/%D0%A1%D0%9F%D0%98%20-%20%D0%A1%D0%BE%D0%B2%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5%20%D0%BF%D1%80%D0%BE%D0%B1%D0%BB%D0%B5%D0%BC%D1%8B%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B8%20(%D0%A4%D0%B8%D0%BE%D0%BD%D0%BE%D0%B2)/labs_realisation/lab_1-3