**Name: VEN THON**
**ID: e20191250**
**Group: I5-GIC-C**

## Assignment Lesson07

1) Find encoder and decoder of LZ77? If we have:
Input string: " abdcaedbdcecabbdeacb" (first block = 7 and second block = 5)
We have string **abdcaedbdcecabbdeacb**

Encoder:

**Step1:**

❖ Compare 5 character from first block with second block

a b d c a e d b d c e c a b b d e a c b

- "a b d c a" ≠ "b d c e c" → move 1 character from the first block
- "b d c a e" ≠ "b d c e c" → move 1 character from the first block
- "d c a e d" ≠ "b d c e c" → move 1 character from the first block

❖ Compare 4 character from first block with second block

a b d c a e d b d c e c a b b d e a c b

- "a b d c" ≠ "b d c e" → move 1 character from the first block
- "b d c a" ≠ "b d c e" → move 1 character from the first block
- "d c a e" ≠ "b d c e" → move 1 character from the first block
- "c a e d" ≠ "b d c e" → move 1 character from the first block

❖ Compare 3 character from first block with second block

a b d c a e d b d c e c a b b d e a c b

- "a b d" ≠ "b d c" → move 1 character from the first block
- "b d c" ≠ "b d c" → match

a b d c a e d b d c e c a b b d e a c b
7 6 5 4 3 2 1
Codeword<6,3, C(e)> (n=3)

**Step2:** move **n+1 (3+1=4)** window at first block
Keep taking 7 characters from first block and 5 characters from second block

**a b d c a e d b d c e** c a b b d e a c b

❖ Compare 5 characters from first block with second block.

**a e d b d** c e **c a b b d** e a c b

- "aedbd" ≠ "cabbd" → move 1 character from first block.
- "edbdc" ≠ "cabbd" → move 1 character from first block.
- "dbdce" ≠ "cabbd" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 4 characters from second block: "cabb".

❖ Compare 4 characters from first block with second block.

**a e d b** d c e **c a b b** d e a c b

- "aedb" ≠ "cabb" → move 1 character from first block.

- “edbd” ≠ “cabb” → move 1 character from first block.
- “dbdc” ≠ “cabb” → move 1 character from first block.
- “bdce” ≠ “cabb” → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 3 characters from second block: “cab”.

❖ Compare 3 characters from first block with second block.

**a e d** b d c e **c a b** b d e a c b

- “aed” ≠ “cab” → move 1 character from first block.
- “edb” ≠ “cab” → move 1 character from first block.
- “dbd” ≠ “cab” → move 1 character from first block.
- “bdc” ≠ “cab” → move 1 character from first block.
- “dce” ≠ “cab” → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 2 characters from second block: “ca”.

❖ Compare 2 characters from first block with second block.

**a e** d b d c e **c a** b b d e a c b

- “ae” ≠ “ca” → move 1 character from first block.
- “ed” ≠ “ca” → move 1 character from first block.
- “db” ≠ “ca” → move 1 character from first block.
- “bd” ≠ “ca” → move 1 character from first block.
- “dc” ≠ “ca” → move 1 character from first block.
- “ce” ≠ “ca” → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 1 character1 from second block: “c”.

❖ Compare 1 character1 from first block with second block.

**a** e d b d c e **c** a b b d e a c b

- “a” ≠ “c” → move 1 character from first block.
- “e” ≠ “c” → move 1 character from first block.
- “d” ≠ “c” → move 1 character from first block.
- “b” ≠ “c” → move 1 character from first block.
- “d” ≠ “c” → move 1 character from first block.
- “c” = “c” → match

a e d b d c e c a b b d e a c b

7 6 5 4 3 2 1

We get: Codeword<2, 1, C(a)> (n=1)

**Step3:** move **n+1 (1+1=2)** window at first block

Keep taking 7 characters from first block and 5 characters from second block

**a e d b d c e c a** b b d e a c b

- Compare 5 characters from first block with second block.

  **d b d c e** c a **b b d e a** c b

  - “dbdce” ≠ “bbdea” → move 1 character from first block.
  - “bdcec” ≠ “bbdea” → move 1 character from first block.
  - “dceca” ≠ “bbdea” → no more character from first block.
  - So, remove 1 character at the end from second block.
  - It rests 4 characters from second block: “bbde”.
- Compare 4 characters from first block with second block.

  **d b d c** e c a **b b d e** a c b

  - “dbdc” ≠ “bbde” → move 1 character from first block.
  - “bdce” ≠ “bbde” → move 1 character from first block.
  - “dcec” ≠ “bbde” → move 1 character from first block.
  - “ceca” ≠ “bbde” → no more character from first block.
  - So, remove 1 character at the end from second block.
  - It rests 3 characters from second block: “bbd”.
- Compare 3 character from first block with second block.

  **d b d** c e c a **b b d** e a c b

  - “dbd” ≠ “bbd” → move 1 character from first block.
  - “bdc” ≠ “bbd” → move 1 character from first block.
  - “dce” ≠ “bbd” → move 1 character from first block.
  - “cec” ≠ “bbd” → move 1 character from first block.
  - “eca” ≠ “bbd” → no more character from first block.
  - So, remove 1 character at the end from second block.
  - It rests 2 characters from second block: “bb”.
- Compare 2 character from first block with second block.

  **d b** d c e c a **b b** d e a c b

  - “db” ≠ “bb” → move 1 character from first block.
  - “bd” ≠ “bb” → move 1 character from first block.
  - “dc” ≠ “bb” → move 1 character from first block.
  - “ce” ≠ “bb” → move 1 character from first block.

- "ec" ≠ "bb" → move 1 character from first block.
- "ca" ≠ "bb" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 1 character from second block: "b".

❖ Compare 1 character from first block with second block.

**d** b d c e c a **b** b d e a c b

- "d" ≠ "b" → move 1 character from first block.
- "b" = "b" → match.

d b d c e c a b b d e a c b

7 6 5 4 3 2 1

We get: Codeword<6, 1, C(b)> (n=1)

**Step4**: move **n+1 (1+1=2)** window at first block

Keep taking 7 characters from first block and 5 characters from second block

**d b d c e c a b b** d e a c b

❖ Compare 5 characters from first block with second block.

**d c e c a** b b **d e a c b**

- "dceca" ≠ "deacb" → move 1 character from first block.
- "cecab" ≠ "deacb" → move 1 character from first block.
- "ecabb" ≠ "deacb" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 4 characters from second block: "deac".

❖ Compare 4 characters from first block with second block.

**d c e c** a b b **d e a c** b

- "dcec" ≠ "deac" → move 1 character from first block.
- "ceca" ≠ "deac" → move 1 character from first block.
- "ecab" ≠ "deac" → move 1 character from first block.
- "cabb" ≠ "deac" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 3 characters from second block: "dea".

❖ Compare 3 character from first block with second block.

**d c e** c a b b **d e a** c b

- "dce" ≠ "dea" → move 1 character from first block.
- "cec" ≠ "dea" → move 1 character from first block.
- "eca" ≠ "dea" → move 1 character from first block.
- "cab" ≠ "dea" → move 1 character from first block.

- "abb" ≠ "dea" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 2 characters from second block: "de".

❖ Compare 2 character from first block with second block.

**d c** e c a b b **d e** a c b

- "dc" ≠ "de" → move 1 character from first block.
- "ce" ≠ "de" → move 1 character from first block.
- "ec" ≠ "de" → move 1 character from first block.
- "ca" ≠ "de" → move 1 character from first block.
- "ab" ≠ "de" → move 1 character from first block.
- "bb" ≠ "de" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 1 character from second block: "d".

❖ Compare 2 character from first block with second block.

**d** c e c a b b **d** e a c b

- "d" = "d" → match.

d c e c a b b d e a c b

7 6 5 4 3 2 1

We get: Codeword<7, 1, C(e)> (n=1)

**Step 5:** move **n+1 (1+1=2)** window at first block

Keep taking 7 characters from first block, but second block rests only 3 characters, so we take only 3 from second block

**d c e c a b b d e** a c b

❖ Compare 3 characters from first block with second block.

**e c a** b b d e **a c b**

- "eca" ≠ "acb" → move 1 character from first block.
- "cab" ≠ "acb" → move 1 character from first block.
- "abb" ≠ "acb" → move 1 character from first block.
- "bbd" ≠ "acb" → move 1 character from first block.
- "bde" ≠ "acb" → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 2 characters from second block: "ac".

❖ Compare 2 characters from first block with second block.

**e c** a b b d e **a c** b

- "ec" ≠ "ac" → move 1 character from first block.

- “ca” ≠ “ac” → move 1 character from first block.
- “ab” ≠ “ac” → move 1 character from first block.
- “bb” ≠ “ac” → move 1 character from first block.
- “bd” ≠ “ac” → move 1 character from first block.
- “de” ≠ “ac” → no more character from first block.
- So, remove 1 character at the end from second block.
- It rests 1 character from second block: “a”.

❖ Compare 1 characters from first block with second block.

**e** c a b b d e **a** c b

- “e” ≠ “a” → move 1 character from first block.
- “c” ≠ “a” → move 1 character from first block.
- “a” = “a” → match.

e c a b b d e a c b

7 6 5 4 3 2 1

We get: Codeword<5, 1, C(c)> (n=1)

**Step 6:** move **n+1 (1+1=2)** window at first block

Keep taking 7 characters from first block, but second block rests only 1 character, so we take only 1 from second block

**e c a b b d e a c** b

❖ Compare 3 characters from first block with second block.

**a** b b d e a c **b**

- “a” ≠ “b” → move 1 character from first block.
- “b” = “b” → match.

a b b d e a c b

7 6 5 4 3 2 1

We get: Codeword<6, 1, null> (n=1)

➢ Because no more character in the second block, we stop here.

- Encode: {<6, 3, C(e)>, <2, 1, C(a)>, <6, 1, C(b)>, <7, 1, C(e)>, <5, 1, C(c)>, <6, 1, null>}
- Result: {“abdcaed”, <6, 3, C(e)>, <2, 1, C(a)>, <6, 1, C(b)>, <7, 1, C(e)>, <5, 1, C(c)>, <6, 1, null>}

Decoder:

**Step 1:** we have to write the first block string

- We get: "abdcaed"
- Use first result of encoder: <6, 3, C(e)>
- Give index from 1 as in the encoder:

a b d c a e d

7 6 5 4 3 2 1

a b d c a e d **b d c**

7 6 5 4 3 2 1

a b d c a e d **b d c e**

7 6 5 4 3 2 1

**Step 2:** move **n+1 (3+1=4)** window

a b d c a e d b d c e (result from step 1)

**a b d c** a e d b d c e

- Use second result of encoder: <2, 1, C(a)>

**a b d c** a e d b d c e

7 6 5 4 3 2 1

**a b d c** a e d b d c e **c**

7 6 5 4 3 2 1

**a b d c** a e d b d c e **c a**

7 6 5 4 3 2 1

**Step 3:** move **n+1 (1+1=2)** window

a b d c a e d b d c e c a (result from step 2)

**a b d c a e** d b d c e c a

- Use third result of encoder: <6, 1, C(b)>

**a b d c a e** d b d c e c a

7 6 5 4 3 2 1

**a b d c a e** d b d c e c a **b**

7 6 5 4 3 2 1

**a b d c a e** d b d c e c a **b b**

7 6 5 4 3 2 1

**Step 4:** move **n+1 (1+1=2)** window

a b d c a e d b d c e c a b b (result from step 3)

**a b d c a e d b** d c e c a b b

- Use forth result of encoder: <7, 1, C(e)>

a b d c a e d b d c e c a b b

7 6 5 4 3 2 1

a b d c a e d b d c e c a b b **d**

7 6 5 4 3 2 1

a b d c a e d b d c e c a b b **d e**

7 6 5 4 3 2 1

**Step 5:** move **n+1 (1+1=2)** window

a b d c a e d b d c e c a b b d e (result from step 4)

**a b d c a e d b d c** e c a b b d e

- Use fifth result of encoder: <5, 1, C(c)>

a b d c a e d b d c e c a b b d e

7 6 5 4 3 2 1

a b d c a e d b d c e c a b b d e **a**

7 6 5 4 3 2 1

a b d c a e d b d c e c a b b d e **a c**

7 6 5 4 3 2 1

**Step 6:** move **n+1 (1+1=2)** window

a b d c a e d b d c e c a b b d e a c (result from step 5)

**a b d c a e d b d c e c** a b b d e a c

- Use last result of encoder: <6, 1, null>

a b d c a e d b d c e c a b a d e a c

7 6 5 4 3 2 1

a b d c a e d b d c e c a b a d e a c **b**

7 6 5 4 3 2 1

Decoder: "abdcaedbdcecabbdeacb"

2) Find encoder and decoder of LZ77? If we have:
Input string: " daddacabeacaebccdaabbeacb" (first block = 8 and second block = 6)

Encoder:

**Step 1:** Compare 6 characters from first block with second block

**d a d d a c** a b **e a c a e b** c c d a a b b e a c b

- "daddac" ≠ "eacaeb" → move 1 character from first block
- "addaca" ≠ "eacaeb" → move 1 character from first block
- "ddacab" ≠ "eacaeb" → no more character from first block
- Compare 5 characters → no match: "dacab" ≠ "eacae"

- Compare 4 characters → no match: “acab” ≠ “eaca”
- Compare 3 characters → no match: “cab” ≠ “eac”
- Compare 2 characters → no match: “ab” ≠ “ea”
- Compare 1 characters → no match: “b” ≠ “e”
- Codeword<0, 0, C(e)> (n=0)

**Step 2:** move **n+1 (0+1=1)** window at first block

- Keep taking 8 characters from first block and 6 characters from second block.

  **d a d d a c a** b e **a c a e b c** c d a a b b e a c b

- “addaca” ≠ “acaebc” → move 1 character from first block
- “ddacab” ≠ “acaebc” → move 1 character from first block
- “dacabe” ≠ “acaebc” → no more character from first block
- Compare 5 characters → no match: “dacabe” ≠ “acaeb”
- Compare 4 characters → no match: “dacabe” ≠ “acae”
- Compare 3 characters → match: “aca” = “aca”

  a d d **a c a** b e **a c a** e b c c d a a b b e a c b

  8 7 6 5 4 3 2 1

- Codeword<5, 3, C(e)> (n=3)

**Step 3:** move **n+1 (3+1=4)** window at first block

- Keep taking 8 characters from first block and 6 characters from second block.

  **a d d a c a b e a c** a e **b c c d a a** b b e a c b

- “cabeac” ≠ “bccdaa” → move 1 character from first block
- “abeaca” ≠ “bccdaa” → move 1 character from first block
- “beacae” ≠ “bccdaa” → no more character from first block
- Compare 5 characters → no match: “eacae” ≠ “bccda”
- Compare 4 characters → no match: “acae” ≠ “bccd”
- Compare 3 characters → no match: “cae” ≠ “bcc”
- Compare 2 characters → no match: “ae” ≠ “bc”
- Compare 1 characters → match: “b” = “b”

  c a **b** e a c a e **b** c c d a a b b e a c b

  8 7 6 5 4 3 2 1

- Codeword<6, 1, C(c)> (n=1)

**Step 4:** move **n+1 (1+1=2)** window at first block

Keep taking 8 characters from first block and 6 characters from second block.

**c a b e a c a e** b c **c d a a b b** e a c b

- “beacae” ≠ “cdaabb” → move 1 character from first block

- “eacaeb” ≠ “cdaabb” → move 1 character from first block
- “acaebc” ≠ “cdaabb” → no more character from first block
- Compare 5 characters → no match: “caebc” ≠ “cdaab”
- Compare 4 characters → no match: “aebc” ≠ “cdaa”
- Compare 3 characters → no match: “ebc” ≠ “cda”
- Compare 2 characters → no match: “bc” ≠ “cd”
- Compare 1 characters → match: “c” = “c”

  b e a **c** a e b c **c** d a a b b e a c b

  8 7 6 5 4 3 2 1

- Codeword<5, 1, C(d)> (n=1)

**Step 5:** move **n+1 (1+1=2)** window at first block

Keep taking 8 characters from first block and 6 characters from second block.

**b e a c a e b c** c d **a a b b e a** c b

- “acaebc” ≠ “aabbea” → move 1 character from first block
- “caebcc” ≠ “aabbea” → move 1 character from first block
- “aebccd” ≠ “aabbea” → no more character from first block
- Compare 5 characters → no match: “ebccd” ≠ “aabbe”
- Compare 4 characters → no match: “bccd” ≠ “aabb”
- Compare 3 characters → no match: “ccd” ≠ “aab”
- Compare 2 characters → no match: “cd” ≠ “aa”
- Compare 1 characters → match: “a” = “a”

  **a** c a e b c c d **a** a b b e a c b

  8 7 6 5 4 3 2 1

- Codeword<8, 1, C(a)> (n=1)

**Step 6:** move **n+1 (1+1=2)** window at first block

Keep taking 8 characters from first block and 6 characters from second block.

**a c a e b c c d** a a **b b e a c b**

- “aebccd” ≠ “bbeacb” → move 1 character from first block
- “ebccda” ≠ “bbeacb” → move 1 character from first block
- “bccdaa” ≠ “bbeacb” → no more character from first block
- Compare 5 characters → no match: “ccdaa” ≠ “bbeac”
- Compare 4 characters → no match: “cdaa” ≠ “bbea”
- Compare 3 characters → no match: “daa” ≠ “bbe”
- Compare 2 characters → no match: “aa” ≠ “bb”
- Compare 1 characters → match: “b” = “b”

a e **b** c c d a a **b** b e a c b

8 7 6 5 4 3 2 1

- Codeword<6, 1, C(b)> (n=1)

**Step 7:** move **n+1 (1+1=2)** window at first block

- Keep taking 8 characters from first block, but second block rests only 4 characters, so we take only 4 from second block

**a e b c c d** a a b b **e a c b**

- “bccd” ≠ “eacb” → move 1 character from first block
- “ccda” ≠ “eacb” → move 1 character from first block
- “cdaa” ≠ “eacb” → move 1 character from first block
- “daab” ≠ “eacb” → move 1 character from first block
- “aabb” ≠ “eacb” → no more character from first block
- Compare 3 characters → no match: “abb” ≠ “eac”
- Compare 2 characters → no match: “bb” ≠ “ea”
- Compare 1 characters → no match: “b” ≠ “e”
- Codeword<0, 0, C(e)> (n=0)

**Step 8:** move **n+1 (0+1=1)** window at first block

Keep taking 8 characters from first block, but second block rests only 3 characters, so we take only 3 from second block

**b c c d** a a b b e **a c b**

- “ccd” ≠ “acb” → move 1 character from first block
- “cda” ≠ “acb” → move 1 character from first block
- “daa” ≠ “acb” → move 1 character from first block
- “aab” ≠ “acb” → move 1 character from first block
- “abb” ≠ “acb” → move 1 character from first block
- “bbe” ≠ “acb” → no more character from first block
- Compare 3 characters → no match: “bbe” ≠ “acb”
- Compare 2 characters → no match: “be” ≠ “ac”
- Compare 1 characters → no match: “a” = “a”

c c d **a** a b b e **a** c b

8 7 6 5 4 3 2 1

- Codeword<5, 1, C(c)> (n=1)

**Step 9:** move **n+1 (1+1=2)** window at first block

Keep taking 8 characters from first block, but second block rests only 1 character, so we take only 1 from second block

**d** a a b b e a c **b**

- “d” ≠ “b” → move 1 character from first block
- “a” ≠ “b” → move 1 character from first block
- “a” ≠ “b” → move 1 character from first block
- “b” = “b” → match

d a a **b** b e a c **b**

8 7 6 5 4 3 2 1

- Codeword<5, 1, null> (n=1)

Because no more character in the second block, we stop here.

- Encode: {<0, 0, C(e)>, <5, 3, C(e)>, <6, 1, C(c)>, <5, 1, C(d)>, <8, 1, C(a)>, <6, 1, C(b)>, <0, 0, C(e)>, <5, 1, C(c)>, <5, 1, null>}
- Result: {“daddacab”, <0, 0, C(e)>, <5, 3, C(e)>, <6, 1, C(c)>, <5, 1, C(d)>, <8, 1, C(a)>, <6, 1, C(a)>, <0, 0, C(e)>, <5, 1, C(c)>, <5, 1, null>}

Decoder:

**Step 1:** we have to write the first block string

- We get: “daddacab”
- Use first result of encoder: <0, 0, C(e)>
- Give index from 1 as in the encoder:

d a d d a c a b

8 7 6 5 4 3 2 1

d a d d a c a b **e**

8 7 6 5 4 3 2 1

**Step 2:** move **n+1 (0+1=1)** window

d a d d a c a b e (result from step 1)

**d** a d d a c a b e

- Use second result of encoder: <5, 3, C(e)>

**d** a d d a c a b e

8 7 6 5 4 3 2 1

**d** a d d a c a b e **a c a**

8 7 6 5 4 3 2 1

**d** a d d a c a b e **a c a e**

8 7 6 5 4 3 2 1

**Step 3:** move **n+1 (3+1=4)** window

d a d d a c a b e a c a e (result from step 2) d a d d a c a b e a c a e

- Use third result of encoder: <6, 1, C(c)>

d a d d a c a b e a c a e

8 7 6 5 4 3 2 1

d a d d a c a b e a c a e **b**

8 7 6 5 4 3 2 1

d a d d a c a b e a c a e **b c**

8 7 6 5 4 3 2 1

**Step 4:** move **n+1 (1+1=2)** window

d a d d a c a b e a c a e b c (result from step 3)

**d a d d a c a** b e a c a e b c

- Use forth result of encoder: <5, 1, C(d)>

**d a d d a c a** b e a c a e b c

8 7 6 5 4 3 2 1

**d a d d a c a** b e a c a e b c **c**

8 7 6 5 4 3 2 1

**d a d d a c a** b e a c a e b c **c d**

8 7 6 5 4 3 2 1

**Step 5:** move **n+1 (1+1=2)** window

d a d d a c a b e a c a e b c c d (result from step 4)

**d a d d a c a b e** a c a e b c c d

- Use fifth result of encoder: <8, 1, C(a)>

**d a d d a c a b e** a c a e b c c d

8 7 6 5 4 3 2 1

**d a d d a c a b e** a c a e b c c d **a**

8 7 6 5 4 3 2 1

**d a d d a c a b e** a c a e b c c d **a a**

8 7 6 5 4 3 2 1

**Step 6:** move **n+1 (1+1=2)** window

d a d d a c a b e a c a e b c c d a a (result from step 5)

**d a d d a c a b e a c** a e b c c d a a

- Use sixth result of encoder: <6, 1, C(b)>

**d a d d a c a b e a c** a e b c c d a a

8 7 6 5 4 3 2 1

**d a d d a c a b e a c** a e b c c d a a **b**

8 7 6 5 4 3 2 1

**d a d d a c a b e a c** a e b c c d a a **b b**

8 7 6 5 4 3 2 1

**Step 7:** move **n+1 (1+1=2)** window

d a d d a c a b e a c a e b c c d a a b b (result from step 6)

**d a d d a c a b e a c a e** b c c d a a b b

- Use seventh result of encoder: <0, 0, C(e)>

**d a d d a c a b e a c a e** b c c d a a b b

8 7 6 5 4 3 2 1

**d a d d a c a b e a c a e** b c c d a a b b **e**

8 7 6 5 4 3 2 1

**Step 8:** move **n+1 (0+1=1)** window

d a d d a c a b e a c a e b c c d a a b b e (result from step 7)

**d a d d a c a b e a c a e b** c c d a a b b e

- Use eighth result of encoder: <5, 1, C(c)>, <5, 1, null>

**d a d d a c a b e a c a e b** c c d a a b b e

8 7 6 5 4 3 2 1

**d a d d a c a b e a c a e b** c c d a a b b e **a**

8 7 6 5 4 3 2 1

**d a d d a c a b e a c a e b** c c d a a b b e **a c**

8 7 6 5 4 3 2 1

**Step 9:** move **n+1 (1+1=2)** window

d a d d a c a b e a c a e b c c d a a b b e a c (result from step 7)

**d a d d a c a b e a c a e b c c** d a a b b e a c

- Use eighth result of encoder: <5, 1, null>

**d a d d a c a b e a c a e b c c** d a a b b e a c

8 7 6 5 4 3 2 1

**d a d d a c a b e a c a e b c c** d a a b b e a c **b**

8 7 6 5 4 3 2 1

- Decoder: “daddacabeacaebccdaabbeacb”