Name: Rotha Dapravith

ID: e20190915

Group: I5-GIC(B)

**Assignment Lesson 7**

1) Find encoder and decoder of LZ77? If we have:

Input string: “ abdcaedbdcecabbdeacb” (first block = 7 and second block = 5)

2) Find encoder and decoder of LZ77? If we have:

Input string: “ daddacabeacaebccdaabbeacb” (first block = 8 and second block = 6)

**Answers:**

1). Find encoder and decoder of LZZ77

We have Input string: “abdcaedbdcecabbdeacb”

* abdcaedbdcecabbdeacb
  + step 1: compare 5 from first block with second block

- abdcaedbdcecabbdeacb

+ abdca ≠ bdcec  move 1 character from first block

+ bdcae ≠ bdcec  move 1 character from first block

+ dcaed ≠ bdcec  no more character from first block

+ So, remove 1 character at the end from second block

+ it rests 4 characters from the second block: “bdce”.

* Step 1 (cont): compare 4 from first block with second block:

- abdcaedbdcecabbdeacb

+ abdc ≠ bdce  move 1 character from first block

+ bdca ≠ bdce  move 1 character from first block

+ dcae ≠ bdce  move 1 character from first block

+ caed ≠ bdce  no more character from first block

+ So, remove 1 character at the end from second block

+ it rests 3 characters from the second block: “bdc”.

* Step 1 (cont): compare 4 from first block with second block:
* abdcaedbdcecabbdeacb

+ abd ≠ bdc  move 1 character from first block

+ bdc = bdc  match or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

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

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<7, 3, C(e)> (n=length=3)
  + Step 2: move n+1 (3+1=4) window at first block
* abdcaedbdcecabbdeacb
  + Compare 5 character from first block and second block
* aedbdcecabbdeacb

+ 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 the second block: “cabb”.

* + Compare 4 character from first block and second block
* aedbdcecabbdeacb

+ 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 the second block: “cab”.

* + Compare 3 character from first block and second block
* aedbdcecabbdeacb

+ 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 3 characters from the second block: “ca”.

* + Compare 2 character from first block and second block
* aedbdcecabbdeacb

+ 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 3 characters from the second block: “c”.

* + Compare 1 character from first block and second block
* aedbdcecabbdeacb

+ 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 or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

a e d b d c e c a b b d e a c b 7 6 5 4 3 2 1

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<2, 1, C(a)> (n=length=1)
  + Step 3: move n+1 (1+1=2) window at first block
* aedbdcecabbdeacb
  + Compare 5 character from first block and second block
* dbdcecabbdeacb

+ dbdce ≠ bbdea  move 1 character from first block.

+ edbdc ≠ bbdea  move 1 character from first block.

+ dbdce ≠ bbdea  no more character from first block.

+ So, By the string remove 1 character at the end from second block is not possible for match and for better let skip and remove 4 characters

+ it rests 1 character from the second block: “b”.

* + Compare 1 character from first block and second block
* dbdcecabbdeacb

+ d ≠ b  move 1 character from first block

+ b = b  match or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

d b d c e c a b b d e a c b 7 6 5 4 3 2 1

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<6, 1, C(b)> (n=length=1).
  + Step 4: move n+1 (1+1=2) window at first block
* dbdcecabbdeacb
  + Compare 5 character from first block and second block
* dcecabbdeacb

+ 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, By the string remove 1 character at the end from second block is not possible for match and for better let skip and remove 4 characters

+ it rests 1 characters from the second block: “d”.

* + Compare 1 character from first block and second block
* dcecabbdeacb

+ d = d  match or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

d c e c a b b d e a c b 7 6 5 4 3 2 1

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<7, 1, C(e)> (n=length=1).
  + Step 5: move n+1 (1+1=2) window at first block
* dcecabbdeacb
  + Compare 3 character from first block and second block
* ecabbdeacb

+ 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, By the string remove 1 character at the end from second block is not possible for match and for better let skip and remove 4 characters

+ it rests 1 characters from the second block: “a”.

* + Compare 1 character from first block and second block
* ecabbdeacb

+ e ≠ a  move 1 character from first block

+ c ≠ a  move 1 character from first block

+ a = a  match or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

e c a b b d e a c b 7 6 5 4 3 2 1

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<5, 1, C(c)> (n=length=1).
  + Step 6: move n+1 (1+1=2) window at first block
* ecabbdeacb
  + Compare 1 character from first block and second block
* abbdeacb

+ a ≠ b  move 1 character from first block

+ b = b  match or equal each other

+ Start to give index in the first block (7 character) from 1 by reversing from the end to the beginning.

a b b d e a c b 7 6 5 4 3 2 1

* Formula Codeword<position, length, C(x)> (x is next character from the last matching character in second block)
* Therefore, we get Codeword<6, 1, null> (n=length=1).
* Because there is no more character in second block, we stop here:
  + Encode: {<7, 3, C(e)>, <2, 1, C(a)>, <6, 1, C(b)>, <7, 1, C(e)>, <5, 1, C(c)>, <6,

1, null>}

* + Result = {“abdcaed”, <7, 3, C(e)>, <2, 1, C(a)>, <6, 1, C(b)>, <7, 1, C(e)>,

<5, 1, C(c)>, <6, 1, null>}

* Decode part:
  + So, we get: “abdcaed” and

Encode: { <7, 3, C(e)>, <2, 1, C(a)>, <6, 1, C(b)>, <7, 1, C(e)>,

<5, 1, C(c)>, <6, 1, null>}

* + Give index from 1 as in encoder: <7, 3, C(e)> 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 the 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 a

* + - 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 the 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 b

* + - 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 the fourth 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 e

* + - 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 the 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 c

* + - 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 the last result of encoder: <6, 1, null> 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

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

* + Decoder: “abdcaedbdcecabbdeacb”

2). Find encoder and decoder of LZ77? If we have:

Input string: “daddacabeacaebccdaabbeacb” (first block = 8 and second block = 6)

* Encode part daddacabeacaebccdaabbeacb
* Step 1: Compare 6 character first block with second block daddacabeacaebccdaabbeacb

+ daddac ≠ eacaeb  Move 1 character from the first block

+ addaca ≠ eacaeb  Move 1 character from the first block

+ ddacab ≠ eacaeb  No more character from first block

* Compare 5 character first block with second block  No match
* Compare 4 character first block with second block  No match
* Compare 3 character first block with second block  No match
* Compare 2 character first block with second block  No match
* Compare 1 character first block with second block  No match “e”≠“b”
  + Codeword <0, 0, C(e)>
    - Step 2: move n+1 (0+1=1) window at first block daddacabeacaebccdaabbeacb
    - Compare 6 character first block with second block  No match
    - Compare 5 character first block with second block  No match
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  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)>
    - Step 3: move n+1 (3+1=4) window at first block addacabeacaebccdaabbeacb
    - Compare 6 character first block with second block  No match
    - Compare 5 character first block with second block  No match
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  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)>
    - Step 4: move n+1 (1+1=2) window at first block cabeacaebccdaabbeacb
    - Compare 6 character first block with second block  No match
    - Compare 5 character first block with second block  No match
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  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)>
    - Step 5: move n+1 (1+1=2) window at first block beacaebccdaabbeacb
    - Compare 6 character first block with second block  No match
    - Compare 5 character first block with second block  No match
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  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)>
    - Step 6: move n+1 (1+1=2) window at first block acaebccdaabbeacb
    - Compare 6 character first block with second block  No match
    - Compare 5 character first block with second block  No match
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  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)>
    - Step 7: move n+1 (1+1=2) window at first block aebccdaabbeacb
    - Compare 4 character first block with second block  No match
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  No match b ≠ e
  + Codeword<0, 0, C(e)>
    - Step 8: move n+1 (0+1=1) window at first block bccdaabbeacb
    - Compare 3 character first block with second block  No match
    - Compare 2 character first block with second block  No match
    - Compare 1 character first block with second block  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)>
    - Step 9: move n+1 (1+1=2) window at first block ccdaabbeacb
    - Compare 1 character first block with second block  Match b = b

d a a b b e a c b 8 7 6 5 4 3 2 1

* + Codeword<5, 1, null>

Because there is no more character in second block, we stop here:

* + Encoder: {<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(b)>, <0, 0, C(e)>, <5, 1, C(c)>, <5, 1, null>}

* Decode part:
* So, we get: “daddacab” and

Encoder: {<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>}

* Step 1: first index encoder: <0, 0, C(e)> 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 and 2nd index encoder: <5, 3, C(e)> 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 and 3rd index encoder: <6, 1, C(c)> 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 and 4th index encoder: <5, 1, C(d)> 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 and 5th index encoder: <8, 1, C(a)> 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 and 6th index encoder: <6, 1, C(b)> 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 and 7th index 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 e

8 7 6 5 4 3 2 1

* Step 8: move n+1 (0+1=1) window and 8th index encoder: <5, 1, C(c)> 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 (0+1=1) window and 9th index 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 b EOF

8 7 6 5 4 3 2 1

* + Decoder: “daddacabeacaebccdaabbeacb”