השוואה בין דחיסת האפמן לדחיסת למפל זיו עבור שתי דוגמאות:

נשתמש בפונקציה compare_lz_huffman (ניתן למצוא אותה בקובץ LZ.py) להשוואה בין שתי השיטות.

```
def compare_lz_huffman(text):
  print("Huffman compression ratio:", \
    Huffman.compression_ratio(text,text+asci))
  print("LZ compression ratio:", lz_ratio(text))
```

הפונקציה מקבלת מחרוזת text ומדפיסה את יחס הדחיסה שמתקבל מדחיסה בכל אחת מהשיטות. עבור האפמן הקורפוס יהיה text+asci, כאשר asci היא המחרוזת המתקבלת מהפקודה הבאה (מייצגת שישור של כל תווי ה ascii):

```
>>> asci = str.join("",[chr(c) for c in range(128)])
```

1. נייצר מחרוזת אקראית rand באופן הבא:

```
>>> asci = str.join("",[chr(c) for c in range(128)])
>>> rand = "".join([random.choice(asci) for i in range(1000)])
>>> rand[0:100]
"nZ_)\x0c1D\x06!4.#i\x14\n5\x1db[\x02o\x17u\x1a7W~\x7f\x17/p.
\x03x`)!)m`IP{\x19?%R\x18<5e\x14'iP\x0b\x02v@\x11Yek\x16\n1gI
_v\x08:\x10\x18,K8SO# Ef\x11@R7bc2T\x1d(|F\x1b\x11KZ|"
```

נפעיל עליה את compare_lz_huffman ונקבל:

```
>>> compare_lz_huffman(rand)
Huffman compression ratio: 0.9907142857142858
LZ      compression ratio: 1.1428571428571428
```

דחיסת האפמן כמעט ולא תשפר לעומת המיפוי הרגיל של תווים לקוד ascii המתאים להם (בן 7 ביטים). הסיבה היא שבמחרוזת האקראית שכיחויות התווים השונים יהיו כמעט זהות (המחרוזת האקראית נבנית ע"י בחירה בהסתברות אחידה של כל תו מתוך 128 תווי ה ascii).
לעומת זאת יחס הדחיסה עבור דחיסת למפל- זיו יהיה כ- 1.1428 ~ 8/7. זאת משום שההסתברות לחזרה באורך 3 או יותר הינה נמוכה מאד כאשר המחרוזת הינה מחרוזת אקראית על פני תווי ה ascii. כלומר בייצוג הביניים לא צפויות להיות מיוצגות חזרות, ולכן הוא צפוי להכיל בהסתברות גבוהה רק תווים בודדים. כל תו בודד מיוצג ע"י 8 ביטים: ביט ראשון הוא 0 ואחריו 7 הביטים המתאימים לקוד ascii של התו.

2. נייצר את המחרוזת s הבאה:

```
>>> s = "".join([chr(ord("a")+i-1)*i for i in range(1,27)])
>>> s
'abbcccddddeeeeeffffffgggggggghhhhhhhhiiiiiiiiijjjjjjjjjjkkkkk
kkkkkkllllllllllllmmmmmmmmmmmmmmnnnnnnnnnnnnnnnnooooooooooooooop
pppppppppppppppqqqqqqqqqqqqqqqqqrrrrrrrrrrrrrrrrrssssssssssss
sssssssstttttttttttttttttttuuuuuuuuuuuuuuuuuuuuvvvvvvvvvvvv
```

```
vvvvvvvvvvwwwwwwwwwwwwwwwwwwwwwwxxxxxxxxxxxxxxxxxxxxxxxxyyyy
yyyyyyyyyyyyyyyyyyyyyzzzzzzzzzzzzzzzzzzzzzzzzzz'
```

נפעיל עליה את `compare_lz_huffman` ונקבל:

```
>>> compare_lz_huffman(s)
Huffman compression ratio: 0.6866096866096866
LZ      compression ratio: 0.2629222629222629
```

אם נסתכל על המילון שמתקבל עבור קוד האפמן, נוכל לראות שאורך הקידוד שקיבל התו "z" קצר מאורך הקידוד שקיבל התו "a" משום ששכיחות התו "z" בקורפוס גבוהה משכיחות התו "a" בקורפוס. (בידקו מהו אורך הקידוד שמתקבל עבור התו "1")
למפל-זיו ידחוס מחרוזת זאת היטב: כל בלוק של אותיות למעט 3 הראשונים: `abbccc` יוסיף לייצוג הביניים תו בודד וחזרה אחת. כלומר ייצוג הביניים יראה כך:

```
>>> lz77_compress2(s)
['a', 'b', 'b', 'c', 'c', 'c', 'd', [1, 3], 'e', [1, 4], 'f',
[1, 5], 'g', [1, 6], 'h', [1, 7], 'i', [1, 8], 'j', [1, 9],
'k', [1, 10], 'l', [1, 11], 'm', [1, 12], 'n', [1, 13], 'o',
[1, 14], 'p', [1, 15], 'q', [1, 16], 'r', [1, 17], 's', [1,
18], 't', [1, 19], 'u', [1, 20], 'v', [1, 21], 'w', [1, 22],
'x', [1, 23], 'y', [1, 24], 'z', [1, 25]]
```