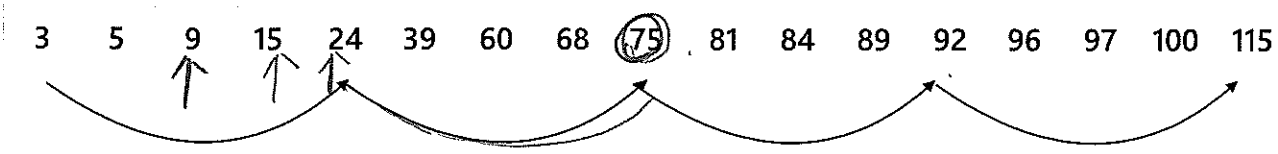
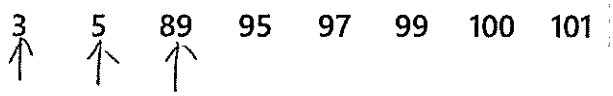
# IR Assignment II

1. Consider a postings intersection between this postings list, with skip pointers:




And the following intermediate result postings list (which has has no skip pointers):




Trace through the postings intersection algorithm.

A. How often is a skip pointer followed (i.e., p1 is advanced to skip(p1))?

Once.

B. How many postings comparisons will be made by this algorithm while intersecting the two lists? (3,3) (5,5) (89,9) (89,15), & 89.

18.

C. How many postings comparisons would be made if the postings lists are intersected without the use of skip pointers?

19

2. We have a two word query. For one term the postings list consist of the following 16 entries.

[ 2, 4, 9, 12, 14, 16, 18, 20, 24, 32, 47, 81, 120, 125, 158, 180 ]

and for the other list it is the one entry postings list

[ 81 ]

Work out how many comparisons would be done to intersect the two postings list with the following two strategies.

i. Using standard postings list.

12 Steps.

12

ii. Using postings list stored with skip pointers, with the suggested skip length of √P.

7

2 4 9 12 14 16 18 20 24 32 47 81 120 125 158 180

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| c | 1 | **0** | 1 | 2 | 3 | 4 | 5 |
| a | 2 | 1 | **0** | 1 | 2 | 3 | 4 |
| t | 3 | 2 | 1 | **0** | **1** | **2** | **3** |

Given two character strings $s_1$ and $s_2$, the *edit distance* between them is the minimum number of *edit operations* required to transform $s_1$ into $s_2$.

4.

**4.A) Write down the entries in the permuterm index dictionary that are generated by the term "mama".**

`m a m a $`

`a m a $ m`

`m a $ m a`

`a $ m a m`

`$ m a m a`

**4.B) If you wanted to search for s*ng in a permuterm wildcard index, what key(s) would one do the lookup on?**

`n g $ s *`

**4.C) Explain with an example post filtering step.**

In a post filtering step, the terms enumerated by the Boolean query on the K-gram index are checked individually against the original query.

For example if the original query string was `red*`; The boolean query mapped to this will lead to matching $re and red to the 3-gram indices. his leads to a match on terms such as retired, which contain the conjunction of the two 3- grams $re and red, yet do not match the original wildcard query `red*`.

In the post-filtering step, the terms enumerated by the Boolean query on the 3-gram index are matched against original query `red*`. This is a simple string-matching operation and weeds out terms such as retired that do not match the original query.