Python 3.6

Anta at Python sine ordbøker (`dict`), frem til versjon 3.6, var implementert med linear probing slik det har blitt undervist i IN2010. (Dette er en liten modifikasjon av sannheten).

I desember 2012 kom Raymond Hettinger med et forslag til en endring av den underliggende representasjonen av ordbøker. I forslaget beskriver Hettinger at den gjeldende representasjonen er unødvendig ineffektiv.

Den organiserer dataene i et array `entries`, der hvert element har plass til en nøkkel k og en verdi v. Arrayet har størrelse N, og mye ledig plass. Hettinger foreslår å omorganisere dataene. Nøkkel-verdi-parene bør i stedet lagres i et tettpakket, dynamisk array `entries` med n elementer, der n er antall nøkkel-verdi-par. Elementene i `entries` kan refereres til fra et annet array `indices` med N elementer (som fremdeles har mye ledig plass).

For eksempel, følgende ordbok:

```
d = {}
d["timmy"] = "red"
d["barry"] = "green"
d["guido"] = "blue"
```

var tidligere representert slik:

```
entries = [(_, _),
```

```
            ("barry", "green"),
            (_, _),
            (_, _),
            (_, _),
            ("timmy", "red"),
            (_, _),
            ("guido", "blue")]
```

der `(_, _)` indikerer en ledig plass. Forslaget er å representere den slik:

```
indices = [_, 1, _, _, _, 0, _, 2]
entries = [("timmy", "red"), ("barry", "green"), ("guido", "blue")]
```

Forslaget over førte til at implementasjonen av ordbøker i Python ble endret fra og med versjon 3.6.

(a) Forklar kort, med naturlig språk, hvordan algoritmene for oppslag og innsetting med linear probing må endres for å ta høyde for den nye strukturen.

(b) Hva er en rehash? Hva er det som forårsaker en rehash?

(c) Hvordan vil en rehash foregå i den nye strukturen?

(d) Hva er fordelene med den nye strukturen?