PYTHON TEST

24. Mart 2018

NAPOMENE

Za potrebe testa napraviti direktorijum **C:\tmp\student\X**, gde je X broj indeksa. Rešenje zadatka treba da se nalazi u tom direktorijumu.

Direktorijum **student\X** zajedno sa rešenjem treba arhivirati i okačiti na lični intranet portal. Direktorijum **C:\tmp\student\X** ne treba brisati.

ZADATAK

1. Napisati funkciju *GetHistogram* koja na osnovu niza ulaznih podataka (karakteri) pravi histogram, tj. računa broj pojavljivanja određenog karaktera. U istom modulu napisati glavni deo programa koji testira funkcionalnost. Primeri ulaznih podataka dostupni su u *snippets.txt*.
   Pseudo kod prototipa funkcije *GetHistogram*:

   ```
   dict GetHistogram(list[])
   ```

2. Implementirati *Priority-queue (max)* strukturu koristeći pomoćne funkcije koje su u pseudokodu prikazane na slici 1. Koristeći implementirane funkcije simulirati rad raspoređivača zasnovanog na prioritetu zadataka: (i) dodati *N* pozitivnih brojeva u red, (ii) redom, element po element, izvaditi *N* elemenata i dodati ih u listu. Proveriti da li su svi elementi sortirani u opadajućem redosledu. U istom modulu napisati glavni deo programa koji testira funkcionalnost.

```
MAX-HEAPIFY(A, i)
1  l = LEFT(i)
2  r = RIGHT(i)
3  if l ≤ A.heap-size and A[l] > A[i]
4       largest = l
5  else largest = i
6  if r ≤ A.heap-size and A[r] > A[largest]
7       largest = r
8  if largest ≠ i
9       exchange A[i] with A[largest]
10      MAX-HEAPIFY(A, largest)
```

```
LEFT(i)
1  return 2i

RIGHT(i)
1  return 2i + 1 [1]
```

```
HEAP-MAXIMUM(A)
1  return A[1]
```

```
MAX-HEAP-INSERT(A, key)
1  A.heap-size = A.heap-size + 1
2  A[A.heap-size] = −∞
3  HEAP-INCREASE-KEY(A, A.heap-size, key)
```

```
HEAP-EXTRACT-MAX(A)
1  if A.heap-size < 1
2       error "heap underflow"
3  max = A[1]
4  A[1] = A[A.heap-size]
5  A.heap-size = A.heap-size − 1
6  MAX-HEAPIFY(A, 1)
7  return max
```

```
HEAP-INCREASE-KEY(A, i, key)
1  if key < A[i]
2       error "new key is smaller than current key"
3  A[i] = key
4  while i > 1 and A[PARENT(i)] < A[i]
5       exchange A[i] with A[PARENT(i)]
6       i = PARENT(i)
```

Slika 1 – Pseudokodovi pomoćnih funkcija Priority-queue (max) strukture indeksirani počev od jedinice.

[1] Za indeksiranje od nule za funkcije Left(i) i Right(i) koristiti 2i+1 i 2(i+1), respektivno.