

# Simulacija raspoređivanja procesa na procesoru prema *Shortest Job First* (SJF) algoritmu sa prioritetom

Napisati konkurentni program koji simulira raspoređivanje procesa na procesoru prema *Shortest Job First* algoritmu s tim da procesi sa višim prioritetom imaju prednost pri zauzimanju procesora. SJF algoritam ima jedan od najboljih pristupa za minimizovanje vremena čekanja, kada procesor unapred zna kolika je količina posla koja je potrebna za izvršavanje svakog procesa.

Procesi su predstavljeni nitima čije je telo funkcija `proces` (data je u fajlu `main.cpp`). Proces se izvršava pozivom metode `izvrsi` klase `Procesor`. Za svaki proces je definisan `id_procesa` koji predstavlja jedinstveni identifikator procesa, `kolicina_posla` kojom je definisano koliko je procesu potrebno vremena da se izvrši (dato u milisekundama) i `prioritet`. Prioritet se zadaje pri kreiranju niti. Prioritet može biti visok ili nizak. Visoki prioritet označen je sa 0, dok je nizak prioritet označen sa 1. Izvršavanje procesa traje onoliko milisekundi koliko je definisano parametrom `kolicina_posla`.

Proces zauzima procesor ukoliko je procesor slobodan. Ukoliko je procesor zauzet, proces čeka na naknadno raspoređivanje od strane raspoređivača. Kada proces zauzme procesor on se izvršava u potpunosti, pre nego što se dešava preključivanje na neki od narednih procesa. Dakle, neće proces nakon svog završetka direktno odrediti koji proces treba da se izvršava sledeći, već to radi posebna nit koja predstavlja raspoređivač. Zadatak procesa, nakon njegovog završetka jeste da obavesti tu drugu nit u kojoj se izvršava raspoređivač, da je potrebno obaviti raspoređivanje (određivanje toga koji će se proces sledeći izvršavati).

Raspoređivanje procesa vrši posebna ciklična nit pozivom funkcije `rasporedjivac`. Ova nit čeka sve dok se ne notifikira da je potrebno da rasporedi sledeći proces. Raspoređivanje se vrši tako da je najpre potrebno obraditi procese sa visokim prioritetom (definisanim kao prioritet 0), a tek naknadno je potrebno obraditi obične procese sa nižim prioritetom (definisanim kao prioritet 1). Prilikom izvršavanja procesa određenog prioriteta potrebno je da procesor zauzme onaj proces koji ima najmanju količinu posla. Ukoliko dva procesa istog prioriteta imaju jednaku količinu posla svedeno je koji od njih će raspoređivač rasporediti.

**Napomena:** prvi proces koji se pokrene naići će na slobodan procesor, pa će ga zauzeti i izvršiti se, bez obzira na pravila o prioritetu i količini preostalog posla. Tek sledeći proces mora biti izabran prema prethodno opisanom algoritmu (SJF).

Primer:

Proces 1: `kolicina_posla`: 200, `prioritet`: 1  
Proces 2: `kolicina_posla`: 100, `prioritet`: 1  
Proces 3: `kolicina_posla`: 500, `prioritet`: 0  
Proces 4: `kolicina_posla`: 800, `prioritet`: 0  
Proces 5: `kolicina_posla`: 900, `prioritet`: 1  
Proces 6: `kolicina_posla`: 900, `prioritet`: 1

Redosled izvršavanja procesa može biti u redosledu : 3 4 2 1 5 6 ili 3 4 2 1 6 5 (može se desiti da se kao prvi izvršeni proces pojavi bilo koji proces – pogledati gornju napomenu).