OOR Ćwiczenie 2 – Równoległość oparta na procesach

Autor: Łukasz Pawłowski

Grupa: 125NCI_B

Link do projektu: https://github.com/mojzesz/oor-rownolegle-cw2

Język programowania: C#

Zadania do zrealizowania:

1. Za pomocą wybranego przez siebie języka programowania zademonstruj tworzenie i współpracę procesów.
2. Zademonstruj działanie kolejki (queue) procesów.

Przykładowe pytania:

- Wymień i omów możliwe stany procesu.
  Nowy (new) – proces jest tworzony. Wykonywany (running) – wykonywane są instrukcje programu. Oczekujący (waiting) – proces oczekuje na jakieś wydarzenie (np. Zwolnienie zasobu). Gotowy (ready) – proces czeka na przydział procesora. Zakończony (terminated) – zakończył działanie i zwalnia zasoby.
- Co to jest zasób? Podaj przykłady.
  Element sprzętowy lub programowy, którego brak może spodowować zablokowanie programu. Taki, który może być przydzielony programowi. Np. Pamięć, czas procesora, drukarka.
- Ile procesów może znajdować się w poszczególnych stanach procesów w danej chwili czasu w systemie z jedną jednostką przetwarzającą (jednym procesorem)?
  Wykonywany może być jeden, reszta może być dowolna, na którą pozwala system operacyjny, planista oraz dostępna pamięć.
- Czym różni się stan procesu gotowy od stanu oczekujący?
  Gotowy może mieć już przydzielone zasoby lub wydarzenie na które czekał już nastąpiło i czeka tylko na przydział procesora.
- Przedstaw klasyfikację zasobów.

Ze względu na sposób wykorzystania (odzyskiwalne – np. RAM, nieodzyskiwalne – np. Czas procesora). Ze względu na sposób odzyskiwania (wywłaszczalne, niewywłaszczalne)

- Wymień i omów rodzaje kolejek procesów.

  Kolejka zadań – wszystkie procesy systemu. Kolejka procesów gotowych – procesy gotowe do działania, przebywające w pamięci głównej. Kolejka do urządzenia – procesy czekające na zakończenie operacji IO. Oczekujących w wyniku synchronizacji z innymi procesami (np. Na semaforze).

- Na czym polega przełączenie kontekstu?

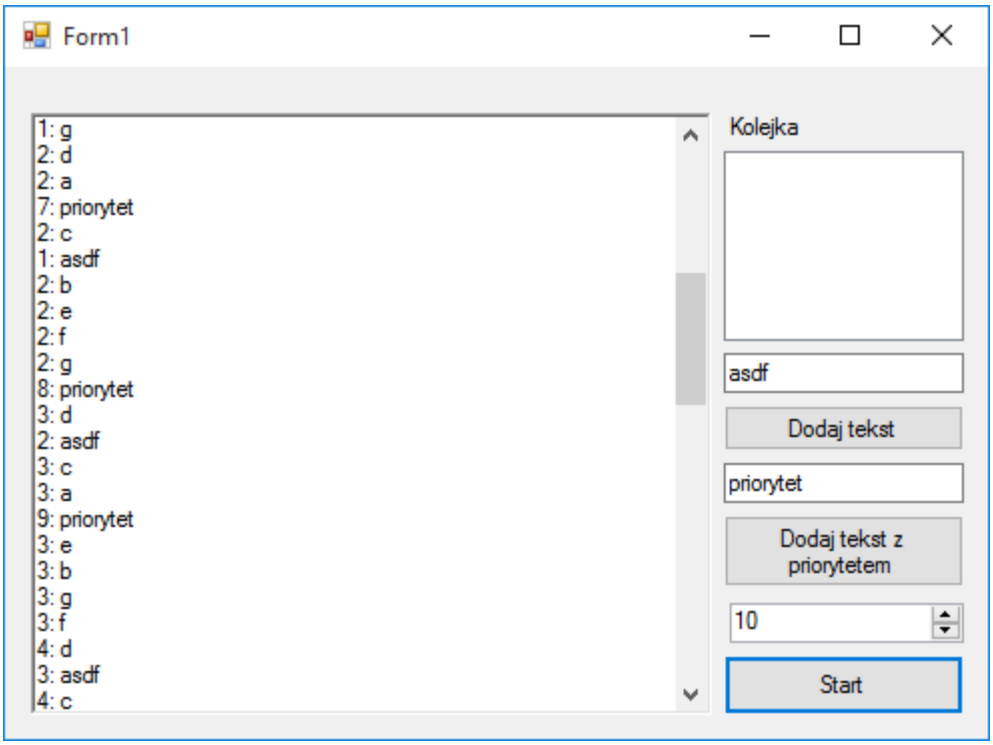
  Podczas gdy wykonywany i potrzebny jest jednen wątek lub proces, pozostałe mogą pozostawać w trybie uśpienia (bezczynności), dopóki nie będą potrzebne (w wyniku akcji użytkownika lub jakiegoś wydarzenia). W takim przypadku do zajścia tej akcji wykonywany jest jeden z wątków/procesów, a pozostałe czekają.

Realizacja zadań

Ad. 1. Oraz Ad. 2.

Kolejne procesy są tworzone poprzez pobranie z listy kolejnego elementu tekstowego i uruchomienie nowego procesu z tym tekstem jako parametrem. Zadaniem procesu jest wykonanie określonej liczby pętli i wypisanie tych tekstów do obiektu RichTextBox. Ponieważ tworzone jest tyle procesów, ile jest elementów na liście i wszystkie one chcą wpisać swój tekst do obiektu RichTextBox, muszą one ze sobą konkurować o dostęp do tego obiektu. Dlatego też planista systemowy ustawia je w kolejce. Program ma możliwość również uruchomienia w każdej chwili procesu z wyższym, niż standardowy priorytetem. Taki proces będzie miał pierwszeństwo w kolejce do procesora nadane przez planistę i – jeśli jest gotowy – wykona się przed innymi procesami.

Na załączonym poniżej obrazku widać, że proces z priorytetem zakończył swoje działanie przed innymi procesami (a,b,c,d,e, f i g)


Form1
1: g
2: d
2: a
7: priorytet
2: c
1: asdf
2: b
2: e
2: f
2: g
8: priorytet
3: d
2: asdf
3: c
3: a
9: priorytet
3: e
3: b
3: g
3: f
4: d
3: asdf
4: c
Kolejka
asdf
Dodaj tekst
priorytet
Dodaj tekst z priorytetem
10
Start