Przemysław Szymoniak

Przetwarzanie współbieżne. Programowanie równoległe i rozproszone.

Sprawozdanie z laboratorium 6.

Celem laboratorium było opanowanie podstaw tworzenia i metod synchronizacji wątków w Javie

W ramach zajęć zrealizowałem następujące kroki:

* Napisałem program, który przy użyciu wątków Javy oblicza histogram dla obrazu **nxm**. Jeden wątek oblicza wystąpienia jednego znaku.

|  |  |
| --- | --- |
| **Kod watek.java**  public class **Watek** extends **Thread**{  private char **znak**;  Obraz **obraz**;  int **moj**\_**id**;  public Watek(int **id**, char **znaczek**, Obraz **obrazek**){ **znak** = **znaczek**; **obraz** = **obrazek**; **moj\_id** = **id**;  }  public void **run**(){ String **wynik** = ""; int **liczbaWystapien**=**obraz**.licz\_wystapienia(**znak**); for(int **i**=**0**; **i**<**liczbaWystapien**;**i**++)  **wynik**+="=";  System.out.println**("Watek "+moj\_id+": "+znak+  " "+wynik+" "**);  }  } | **Kod uruchamiający:**  **Watek**[] **NewThr** = new **Watek**[**num\_threads**];  for (int **i** = **0**; **i** < **num\_threads**; **i**++)  {  (**NewThr**[**i**] = new **Watek**(**i**, (**char**)(**33+i**),**obraz\_1**)).**start**();  }  for (int **i** **= 0**; **i** < **num\_threads**; **i**++) {  try {  **NewThr**[**i**].join();  } catch (InterruptedException **e**) {}  }  Wynik działania kodu dla 4-ch wątków: **(Pełna wersja w załączniku)**  } |

* Napisałem program, który oblicza histogram z podziałem na podzadania.(Przedstawiam różnice w stosunku do kodu powyżej)

|  |  |
| --- | --- |
| **Kod watek.java**  public class **Watek2** extends **Thread** implements **Runnable**  {  Dodatkowo potrzebujemy przechowywać oraz :  int **liczbaZnakow**;  public **Watek2**(**…**, int **lZnakow**){ **… liczbaZnakow**= **lZnakow**; }  Większe zmiany poczyniłem w funkcji run:  **public** **void** **run**() { **for** (**int** **i** = 0; **i** < **liczbaZnakow**; **i**++){  **String** **wynik** = ""; int **liczbaWystapien** = **obraz**.licz\_wystapienia(**znak**);  for (int **j** = 0; **j** < **liczbaWystapien**; **j**++) **wynik** += "=";  System.out.println("**Watek** " + **moj\_id** + "**:** " + **znak** + " " + **wynik** + " ");  **znak**++;  }  }  } | **Kod uruchamiający:**  Dodatkowo potrzebujemy obliczyć:  int **ileZnakow** = **94**/**num**\_**threads**;  int **rekompensata** = **0**;  oraz uwzględnić w pętli tworzącej wątek niepodzielność ilości danych przez liczbe wątków:  for (int **i** = **0**; **i** < **num\_threads**; **i**++) { if(**i+1**==**num\_threads**) **rekompensata**=(**94%num\_threads**); (**NewThr2**[**i**] = new **Watek2**(**i**, (**char**)(**33+i\*ileZnakow**), **obraz\_1**,(**ileZnakow+rekompensata**))).**start**();  }  Wynik działania aplikacji dla 4-ch wątków: **(Pełna wersja w załączniku)** |

Wnioski:

* Pierwsza wersja kodu obliczająca ilość wystąpień  
  jednego znaku nie jest optymalna ponieważ   
  aby przeliczyć występowanie wszystkich symboli wymaga uruchomienia 94 wątków.
* Druga wersja programu jest pod tym względem lepsza, ponieważ możemy wyskalować ilość wątków zależnie od posiadanego procesora a program w każdym przypadku obliczy poprawnie histogram obrazu.
* Wykorzystanie języka programowania Java   
  pozwala na działanie aplikacji niezależnie od środowiska w jakim go uruchamiamy.
* Ilość wątków tworzonych w aplikacji powinna być dostosowana do tego ilu wątkowy jest nasz procesor. Wykorzystanie nadmiernej liczby (jak w I przypadku 94 wątki) może znacząco obniżyć wydajność napisanej przez nas aplikacji.