Koncepcja projektu nr 2

„Szeregowanie procesów”

1. Ogólne rozwiązanie problemu
   1. Wykorzystując jedną kolejkę dla procesów użytkownika.
   2. Przy pojawieniu się nowego procesu, jeśli jest priorytetowy zostaje wrzucony na początek kolejki, w innym wypadku jest umieszczany na końcu. Dostęp do procesora dostaje element pierwszy w kolejce. W momencie otrzymania dostępu do procesora, przez proces priorytetowy kwant czasu zostaje wydłużony (dla tego procesu) do wielkości limitu czasu (stała w kodzie) a sam proces jest zdegradowany do procesu normalnego, jeśli proces nie zakończy swojego działania do upłynięcia limitu czasu zostaje on przeniesiony na koniec kolejki już jako proces normalny. Jeśli nie ma procesów priorytetowych procesy normalne są szeregowane algorytmem Round-Robin.
   3. Użytkownik ustawia priorytet procesu poprzez wywołanie z argumentem, gdzie jeśli argument > 0 wtedy proces jest priorytetowy, a jeśli argument <= 0 wtedy proces jest normalny.
2. Lista modyfikowanych plików:
   1. usr/include/minix:
      1. **callnr.h** – dodanie wywołania, zwiększenie liczby wywołań
      2. **com.h** – dodanie stałej do wywołania
   2. usr/src/mm:
      1. **proto.h** – dodanie prototypu wywołania
      2. **table.c** – dodanie wywołania do tabeli
      3. **main.c** – dodanie funkcji obsługującej wywołanie
   3. usr/src/fs:
      1. **table.c** – dodanie „*no\_sys”* do tabeli
   4. usr/src/kernel:
      1. **proc.h** – dodanie zmiennej odpowiedzialnej za priorytet procesu
      2. **proc.c** – modyfikacja funkcji *ready()* aby realizował zadany algorytm szeregowania
      3. **system.c** – modyfikacja funkcji *do\_fork()* dla procesów klasy USER aby był zainicjowany odpowiedni priorytet, dodanie definicji funkcji *do\_setpri()*
      4. **main**.**c** – zainicjowanie zmiennych
      5. **clock.c** – dodanie obsługi procesów priorytetowych i dodanie stałej limitu czasu
   5. root:
      1. **test.c** – program testujący działanie algorytmu
      2. **1.sh** – skrypt testujący
3. Program testujący utworzy kilka procesów użytkownika (priorytetowych i normalnych w różnej kolejności) następnie wyświetli jakie procesy zostały utworzone i w jakiej kolejności zostały ukończone.