Ek het n proses klas geskep met; die proses naam, die proses tyd en die prioriteit van die proses op ‘n 1-10 skaal as attribute. Met die proses klas alleen, was ek in staat om die kortste werk eerste, die prioriteit en die round robin algoritmes te skep. Om die ‘multiple queues’ algoritme te skryf moes ek nog n klas skep met die naam TypeClass. TypeClass bevat twee attribute, eerstens die tipe proses se naam en tweedens n lys van al die prosesse wat aan die proses tipe behoort.

Elke proses wat deer die GUI ingelees word, word in ‘n lys gestoor. Om die kortste werk eerste algoritme te skep het ek die lys deur ‘n algoritme gesit wat die lys sorteer van kortste proses na langste en daarna deur die lys ge-itereer en elke proses af gevoer. Dieselfde stappe was gevolg toe die prioriteit algoritme geskep is maar in die geval word die lys gesorteer volgens die prioriteit attribuut en daarna af gevoer.

Om die ‘round robin’ algoritme te skep was meer uitdagend. In die geval het ek van ‘n **while** loop gebruik gemaak saam twee **for** loops. Die **while** loop voer af solank daar nog ‘n proses in die lys is wat nog nie volledig af gevoer het nie. Binne die **while** loop is daar ‘n **for** loop wat deur elke proses in die lys itereer. Binne die **for** loop is daar nog n **for** loop wat vyf keer uitvoer. Hy voer vyf keer af aangesien vyf my gekose quantum grote is vir die ‘round robin’ algoritme. Deur konstant boek te hou van die hoeveelheid keer die **for** en **while** loops af voer en die gebruik van ‘n **if** stelling, hou die prosesse wat volledig af gevoer is op af voer en die wat nog nie volledig af gevoer is nie wag sy beurt en voer af tot hy volledig afgevoer is.

Die ‘multiple queue’ algoritme maak gebruik van die typeClass. Daar is vier objekte van die typeClass genaamd primer, sekondêr, game en batch. Die objekte verteenwoordig die tipe prosesse wat ingelees word deur die gebruiker. Die prosesse word by die objekte van sy tipe geplaas en n algoritme breek die prosesse op in hul aktiwiteite en stoor dit in een opvolgende reeks van aktiwiteite. Nou het ek vier objekte wat elkeen ‘n reeks van aktiwiteite besit (die reeks besit al die aktiwiteite van elke proses wat aan die tipe behoort). Ek plaas die objekte in ‘n lys wat gesorteer is in ‘n volgorde van prioriteit, die game objek word eerste af gevoer gevolg met die primêre, dan sekondêre en dan die batch objek se proses aktiwiteite. Ek gebruik ‘n quantum van 20 en ‘n algoritme wat klein bietjie verskil van die een wat gebruik was in die ‘round robin’ algoritme. Die **while** loop itereer tot elke objek se aktiwiteite volledig af gevoer is, ‘n **for** loop binne die **while** itereer deur elke objek in die tipe lys en nog ‘n **for** loop binne die **for** loop hardloop deur twintig quantum’s. Elke quantum verteenwoordig een aktiwiteit wat aan ‘n proses behoort van die tipe objek waar die eerste **for** loop tans is. Dit beteken dat van die prosesse wat aan die game objek behoort voer twintig aktiwiteite af, die aktiwiteite kan aan meer as een proses behoort of net ‘n deel wees van een proses wat meer as twintig aktiwiteite het. As die twintig aktiwiteite (quantum’s) klaar af gevoer het beweeg die **for** loop aan na die volgende objek in die tipe lys wat in die geval die primêre objek is en voer weer twintig aktiwiteite (quantum’s) af. Dit word herhaal tot dat elke aktiwiteit in elke tipe objek volledig af gevoer is.