process scheduling project

## What is the Process Scheduling?

The process scheduling is the manager handles the removal of the running process from the CPU and selects another process on the Ready Queue using a particular strategy, it‘s an important part of Multiprogramming .which allows more processes to be loaded to the memory for execution at the same time and the loaded processes share the CPU by multiplexing.

## What are the Process Scheduling algorithms Types?

* There are four popular process scheduling algorithms:
* First come First serve (FCFS)
* Round Robin
* Priority
* Shortest job first
* Two concepts :
* Pre-emptive:it is mean if there is a process working in CUP and a process arrived the process working In the CPU may be interrupted
* Non-pre-emotive: it means the process that is working on the CPU can’t be interrupted for any reason only for waiting for the I/O device or the end of the execution time.

## What are the project requirements?

* develop a project that takes process information, those processes have to schedule to be executed by the CPU and visualize how the processes are scheduled in the CPU according to a specific algorithm.
* **Inputs:**
* Type of scheduler(Algorithm).
* no of Processes.
* required information about each process according to the scheduler type.
* **Outputs:**
* Timeline showing the order and time taken by each process (Gantt Chart).
* Average waiting time using.

## how the project achieves the requirements?

### Parts of the project

* Design and Implement the GUI.
* Design the logic.

### Gui Design

first of all the GUI is designed to be easy and simple for the user as in figure (1), I developed this gui application using Qt c++.A screenshot of a cell phone

Description automatically generated

**Figure 1: GUI design**

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 3:average waiting time

Figure 2: visualization of processes

#### User Guide

1. The user **should** choose the algorithm.
2. User starts **adding processes data** (Name, Burst Time, Arrival Time, and Priority if any of the Priority algorithms have been chosen).
3. **click Add Process** button to add the process data then it would appear in the text area on the right.
4. Then **add slice time if the round robin has been chosen** and **click the Run button**.
5. Then the average waiting time and the visualization windows of the Processes Scheduling would appear as in figure 2 and figure 3.
6. User can reset the program by **USING the RESET BUTTON** (icon on the top left)for adding new data and having another algorithm.

**AT THE END:**

One of the disadvantages of this project is the graph, which is unclear to the user to know exactly when the process begins and when it ends a that happen cause I used the charts library, but now I learned a better thing to draw this type of chart using graphics view library, as I use it in the second project to draw the memory space.

#### Flow Chart

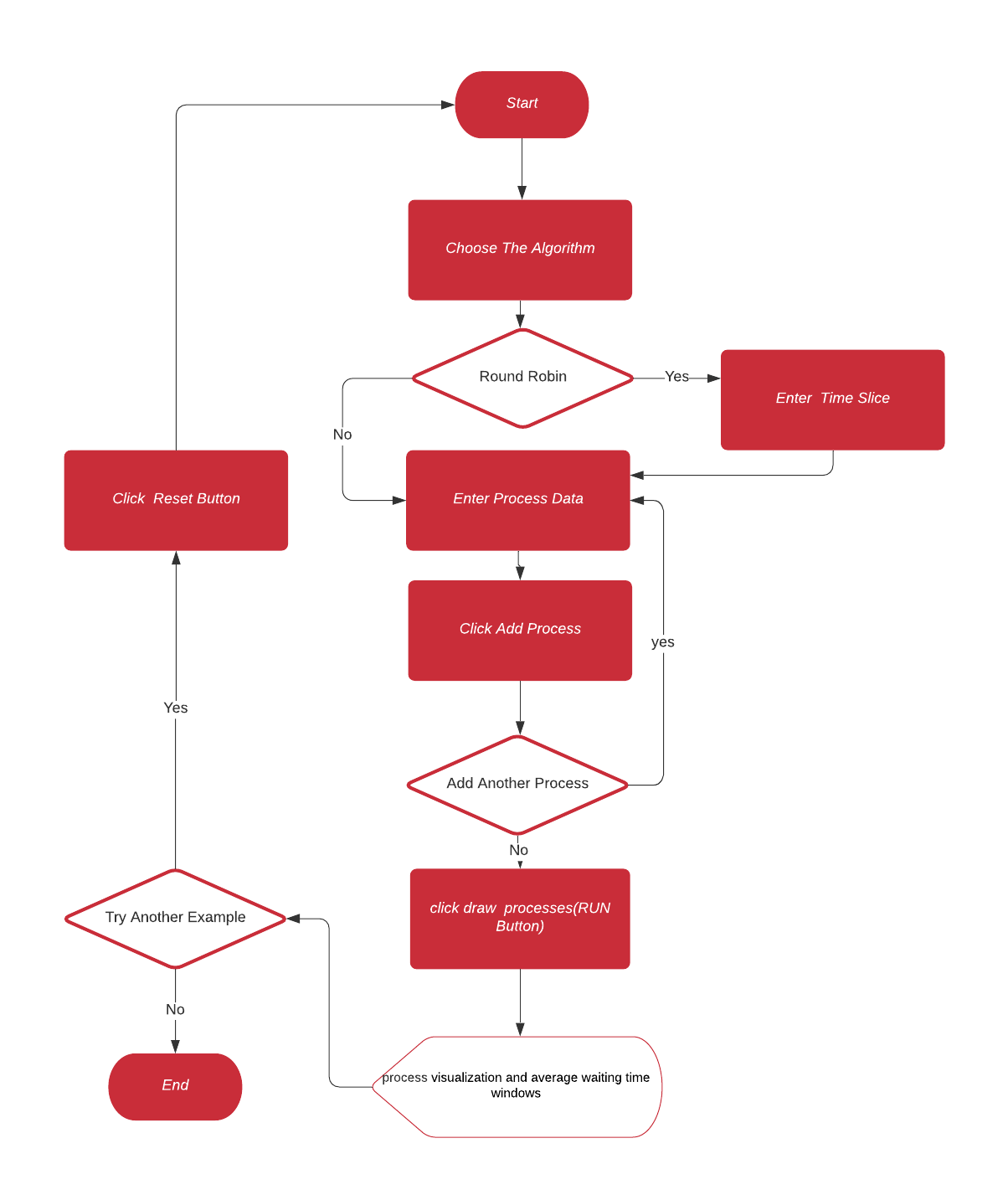


Figure :Flow Chart For Gui Design.

### Algorithms Logic Design

* the project is divided into 6 parts as follows:
* FIRST COME FIRST SERVE.
* ROUND ROBIN.
* PRIORITY PRE-EMPTIVE.
* PRIORITY NON-PRE-EMPTIVE.
* SHORTEST JOB FIRST PRE-EMPTIVE.
* SHORTEST JOB FIRST NON-PRE-EMPTIVE.

#### 1-First come first serve

* This algorithm gives higher priority to the process which arrives earlier, so it is really like the Queue and is implemented using the Queue data structure.
* Now let’s see how our project handles this algorithm :
* We would enter 3 processes (p1 arrived at time 2, p2 arrived a time 3, p3 arrived at time 1, p4 at time 0), and all have the same burst time 4 s.
* A screenshot of a cell phone

  Description automatically generatedour calculation is p4 then p3 then p1 the p2 and the average time is =segma (finish time-arrival time-burst time)for all process/(number of process) = (p1(12-2-4)+p2(16-3-4)+p3(8-1-4)+p4(4-0-4))/4=4.5.

Figure 4:First come first serve (Input)

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 6:First come first serve(AVG Time)

Figure 5:First come first serve(scheduling)

#### 2-Round ROBIN

* This algorithm is really useful for the user interface as it gives each process a slice of time to be executed by the processor which makes the programs more interactive.
* Now let’s see how our project handles this algorithm :
* A screenshot of a cell phone

  Description automatically generatedA screenshot of a cell phone

  Description automatically generatedA screenshot of a cell phone

  Description automatically generatedWe would enter 4 processes (p1 arrived at time 0, p2 arrived a time 0, p3 arrived at time 1, p4 at time 1), and all have the same burst time 4 s with slice time 1 s.

Figure 9:Round ROBIN (AVG Time)

Figure 8: Round ROBIN (Scheduling)

Figure 7:Round ROBIN (Input)

#### 3-PRIORITY PRE-EMPTIVE

* This algorithm executes the processes according to the priority, the lower priority process executes first, but as it is pre-emptive it can interrupt the process which takes the CPU if it has a priority less than the newcomer.
* Now let’s see how our project handles this algorithm :
* We would enter 3 processes (p1 arrived at time 0 PRIORITY 1, p2 arrived a time 1 PRIORITY 0, p3 arrived at time 2 PRIORITY 2 ), and all have the same burst time 4 s.
* our calculation :

at time 0:p1 the only one so would start executing.

at time 1: p2 arrives and has higher PRIORITY so it would take the processor.

at time 2:p3 arrives and it has lower PRIORITY than p2 which executes on the processor so it would wait till p2 finishes then search for the next higher PRIORITY which is p1 than p3.

the average time is =segma (finish time-arrival time-burst time)for all process/(number of process) = (p1(8-0-4)+p2(5-1-4)+p3(12-2-4))/3=3.3333.

A screenshot of a cell phone

Description automatically generated

Figure 10:PRIORITY PRE-EMPTIVE(Input)

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 12:PRIORITY PRE-EMPTIVE(AVG time)

Figure 11:PRIORITY PRE-EMPTIVE(Scheduling)

#### 4-PRIORITY NON-PRE-EMPTIVE

* This algorithm executes the processes according to the priority, the lower priority process executes first, but as it Is non-pre-emptive it can’t interrupt the process which takes the CPU till this process need i/o operation or complete execution.

A screenshot of a cell phone

Description automatically generated

Figure 13:PRIORITY NON PRE-EMPTIVE (Input)

* Ex:

The same previous example

At time 0:p1 execute

At time 1:p2 arrive but it is non-pre-emptive so no interruption p1 working till the end.

At the time 2:p3 arrive p1 still working.

At time 4:p1 finish execution then we are looking for the next higher priority in this case p2 than p3.

* the average time is =segma (finish time-arrival time-burst time)for all process/(number of process) =

(p1(4-0-4)+p2(8-1-4)+p3(12-2-4))/3=3.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 14:PRIORITY NON PRE-EMPTIVE(Scheduling)

Figure 15:PRIORITY NON PRE-EMPTIVE(AVG time)

#### 5-SHORTEST JOP FIRST PRE-EMPTIVE

* This algorithm gives the shortest process higher priority, and as it is pre-emptive the process which on the processor can be interrupted if a shorter one arrived and must be shorter than the remainder burst time of the one on the processor.
* A screenshot of a cell phone

  Description automatically generatedExample:
* **Input:**

P1 arrives at time 0 with burst time 9

p2 arrives at time 2 with burst time 7

p3 arrives at time 5 with burst time 1

* **expected output:**

at time 0: p1 execute

at time 2: (p1 size =7)=(p2 size=7) p1 continue.

at time 5: (p1 size =4)>(p3 size =1)p3 execute.

at time 6:p3 finish (p1 size =4) < (p2 size =7) p1 execute.

at time 10:p2 execute.

Figure 16:SHORTEST JOP FIRST PRE-EMPTIVE(Input)

* A screenshot of a cell phone

  Description automatically generatedthe average time is =segma (finish time-arrival time-burst time) for all process/(number of process) = (p1(10-0-9)+p2(17-2-7)+p3(6-5-1))/3=3.

Figure 18:SHORTEST JOP FIRST PRE-EMPTIVE(AVG time)

A screenshot of a cell phone

Description automatically generated

Figure 17:SHORTEST JOP FIRST PRE-EMPTIVE(Scheduling)

#### 6- SHORTEST JOB FIRST NON PRE-EMPTIVE

* This algorithm gives the shortest process higher priority, but as it Is non-pre-emptive it can’t interrupt the process which takes the CPU till this process need i/o operation or complete execution.
* A screenshot of a cell phone

  Description automatically generatedEx:

Figure 19:SHORTEST JOP FIRST NON PRE-EMPTIVE(Input)

* **Input:**

P1 arrives at time 0 with burst time 9

p2 arrives at time 2 with burst time 7

p3 arrives at time 5 with burst time 1

* **expected output:**

at time 0: p1 execute till end

at time 9: (p2 size =7)>(p3 size=1) p3 execute till ends.

at time 10:p2 execute.

* the average time is =segma (finish time-arrival time-burst time) for all process/(number of process) =(p1(9-0-9)+p2(17-2-7)+p3(10-5-1))/3=4.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Figure 21:SHORTEST JOP FIRST NON PRE-EMPTIVE(AVG time)

Figure 20:SHORTEST JOP FIRST NON PRE-EMPTIVE(Scheduling)

#### Class diagram

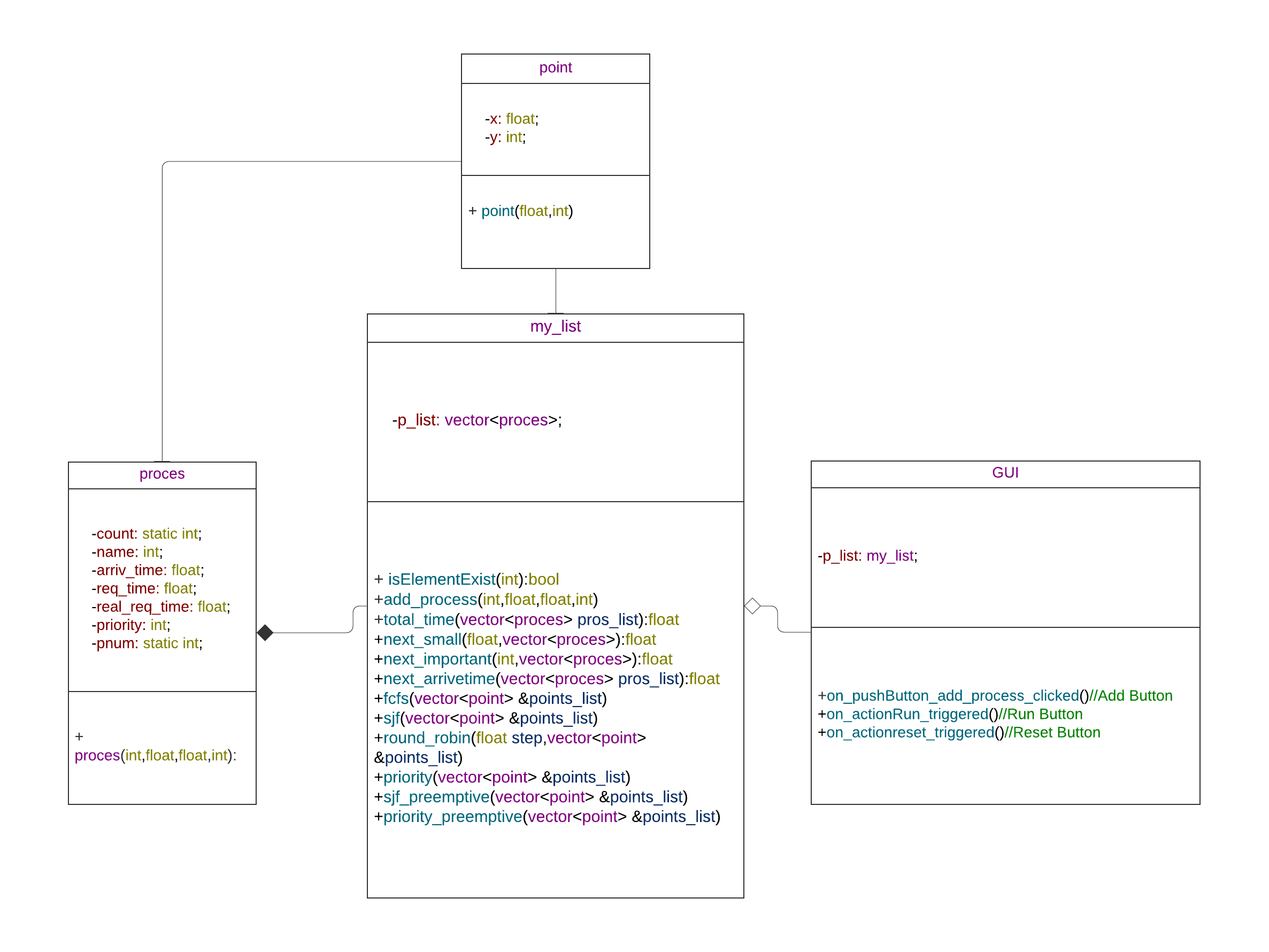


Figure 22:Project Class Diagram

#### Classes Description

* The class process is used to represent the process data.
* The classes point are used to represent the points (start and end for each process) to be drawn in the chart.
* Class my list used to arrange and apply the algorithms on the processes.
* Class GUI responsible for interfacing.