

OPERATING SYSTEM

GUI BASED SIMULATOR ON CPU SCHEDULING ALGORITHMS

G1 TEAM 1

Hardik Inani (Team Leader)	20BCP012
Nilay Patel	20BCP005
Rahul Gulati	20BCP024
Dhvanil Bhagat	20BCP027
Shrey Makadiya	20BCP028
Drashti Bhavsar	20BCP040

TABLE OF CONTENTS

Python Introduction	1
PyQt5 Introduction	2
PyQt5 Components	3
PyQt Introduction	4
Major Classes	5
Using QtDesigner	10
Using the GUI Simulator	13
Contact Us	20

Python

Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available in source or binary form for all major platforms from the Python Website, and may be freely distributed. The same site also contains distributions of and pointers to many free third party Python modules, programs and tools, and additional documentation.

The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications.

PyQt5

Qt is set of cross-platform C++ libraries that implement high-level APIs for accessing many aspects of modern desktop and mobile systems. These include location and positioning services, multimedia, NFC and Bluetooth connectivity, a Chromium based web browser, as well as traditional UI development.

PyQt5 is a comprehensive set of Python bindings for Qt v5. It is implemented as more than 35 extension modules and enables Python to be used as an alternative application development language to C++ on all supported platforms including iOS and Android.

PyQt5 may also be embedded in C++ based applications to allow users of those applications to configure or enhance the functionality of those applications.

PyQt5 Components

PyQt5 comprises a number of different components. First of all there are a number of Python extension modules. These are all installed in the PyQt5 Python package and are described in the list of modules.

Each extension module has a corresponding PEP 484 defined stub file containing type hints for the module's API. This can be used by static type checkers such as mypy.

PyQt5 contains plugins that enable Qt Designer and qmlscene to be extended using Python code. See [Writing Qt Designer Plugins](#) and [Integrating Python and QML](#) respectively for the details.

PyQt5 also contains a number of utility programs.

- `pyuic5` corresponds to the Qt `uic` utility. It converts QtWidgets based GUIs created using Qt Designer to Python code.
- `pyrcc5` corresponds to the Qt `rcc` utility. It embeds arbitrary resources (eg. icons, images, translation files) described by a resource collection file in a Python module.
- `pylupdate5` corresponds to the Qt `lupdate` utility. It extracts all of the translatable strings from Python code and creates or updates `.ts` translation files. These are then used by Qt Linguist to manage the translation of those strings.

The DBus support module is installed as `dbus.mainloop.pyqt5`. This module provides support for the Qt event loop in the same way that the `dbus.mainloop.glib` included with the standard `dbus-python` bindings package provides support for the GLib event loop. The API is described in [DBus Support](#). It is only available if the `dbus-python` v0.80 (or later) bindings package is installed. The `QtDBus` module provides a more Qt-like interface to DBus.

PyQt5 includes a large number of examples. These are ports to Python of many of the C++ examples provided with Qt. They can be found in the `examples` directory of the `sdist`.

Finally, PyQt5 contains the specification files that allow bindings for other Qt based class libraries that further extend PyQt5 to be developed and installed.

PyQt Introduction

PyQt5 also contains a number of utility programs.

- `pyuic5` corresponds to the Qt `uic` utility. It converts QtWidgets based GUIs created using Qt Designer to Python code.
- `pyrcc5` corresponds to the Qt `rcc` utility. It embeds arbitrary resources (eg. icons, images, translation files) described by a resource collection file in a Python module.
- `pylupdate5` corresponds to the Qt `lupdate` utility. It extracts all of the translatable strings from Python code and creates or updates `.ts` translation files. These are then used by Qt Linguist to manage the translation of those strings.

The DBus support module is installed as `dbus.mainloop.pyqt5`. This module provides support for the Qt event loop in the same way that the `dbus.mainloop.glib` included with the standard `dbus-python` bindings package provides support for the GLib event loop. The API is described in [DBus Support](#). It is only available if the `dbus-python v0.80` (or later) bindings package is installed. The `QtDBus` module provides a more Qt-like interface to DBus.

PyQt5 includes a large number of examples. These are ports to Python of many of the C++ examples provided with Qt. They can be found in the `examples` directory of the sdist.

Finally, PyQt5 contains the specification files that allow bindings for other Qt based class libraries that further extend PyQt5 to be developed and installed.

Major Classes

PyQt API is a large collection of classes and methods. These classes are defined in more than 20 modules. Following are some of the frequently used modules: `QtCore`

Core non-GUI classes used by other modules

- **`QtGui`**

Graphical user interface components

- **`QtMultimedia`**

Classes for low-level multimedia programming

- **`QtNetwork`**

Classes for network programming

- **`QtOpenGL`**

OpenGL support classes

- **`QtScript`**

Classes for evaluating Qt Scripts

- **`QtSql`**

Classes for database integration using SQL

- **`QtSvg`**

Classes for displaying the contents of SVG files

- **`QtWebKit`**

Classes for rendering and editing HTML

- **`QtXml`**

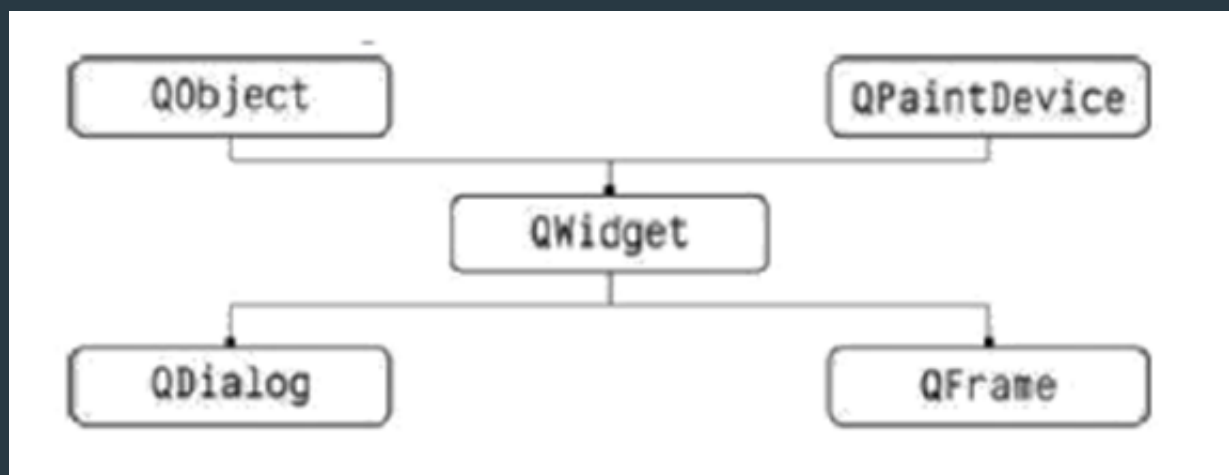
Classes for handling XML

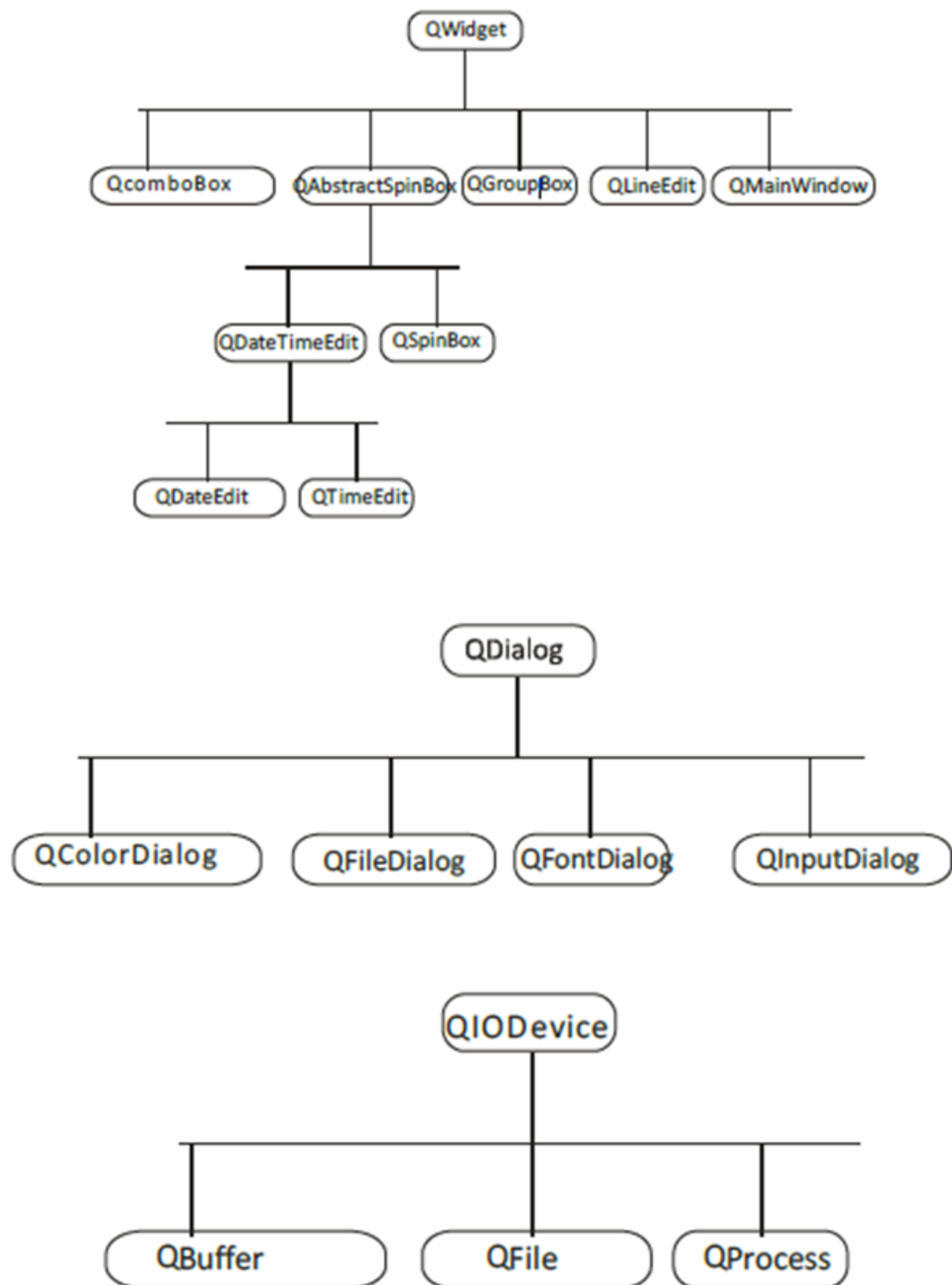
- **`QtAssistant`**

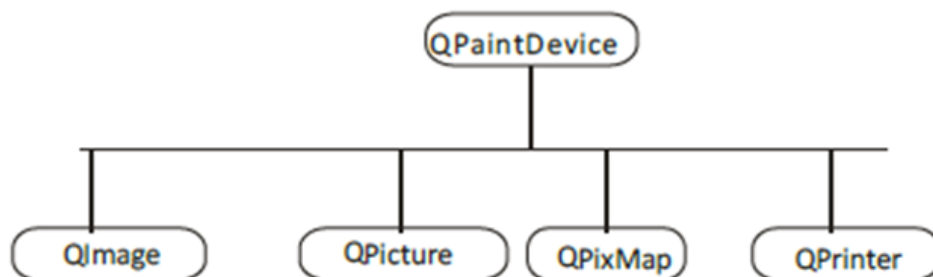
Support for online help

- **`QtDesigner`**

Classes for extending Qt Designer







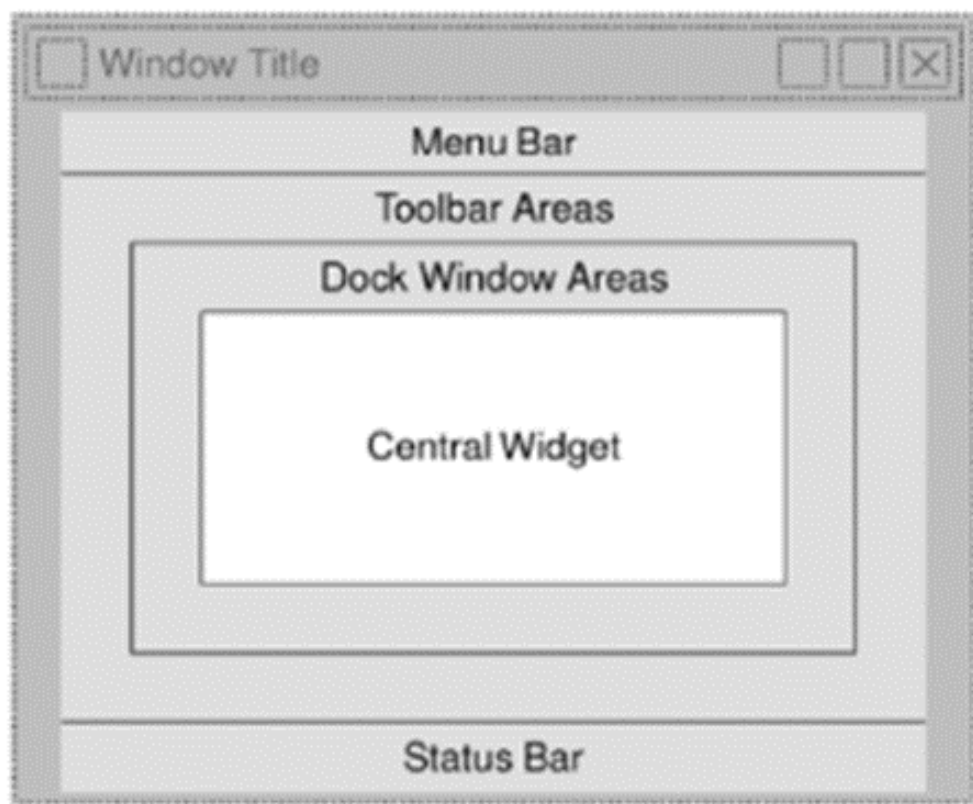
Here is a select list of frequently used widgets:

QLabel	Used to display text or image
QLineEdit	Allows the user to enter one line of text
QTextEdit	Allows the user to enter multi-line text
QPushButton	A command button to invoke action
QRadioButton	Enables to choose one from multiple options
QCheckBox	Enables choice of more than one options
QSpinBox	Enables to increase/decrease an integer value
QScrollBar	Enables to access contents of a widget beyond display aperture
QSlider	Enables to change the bound value linearly.
QComboBox	Provides a dropdown list of items to select from
QMenuBar	Horizontal bar holding QMenu objects
QStatusBar	Usually at bottom of QMainWindow, provides status information.
QToolBar	Usually at top of QMainWindow or floating. Contains action buttons
QListView	Provides a selectable list of items in ListMode or IconMode
QPixmap	Off-screen image representation for display on QLabel or QPushButton object
QDialog	Modal or modeless window which can return information to parent window

A typical GUI based application's top level window is created by **QMainWindow** widget object. Some widgets as listed above take their appointed place in this main window, while others are placed in the central widget area using various layout managers.

The following diagram shows the QMainWindow framework:

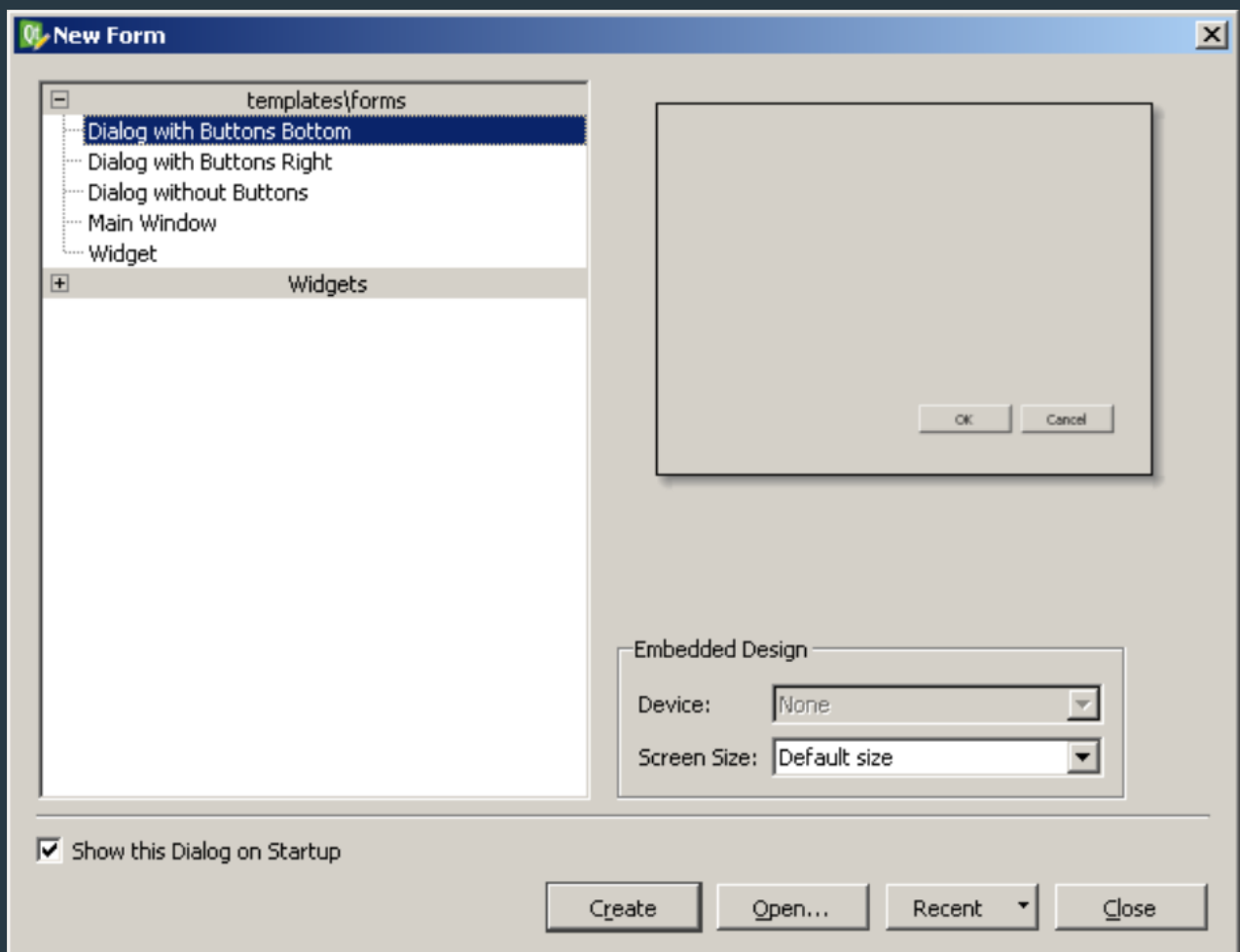
1



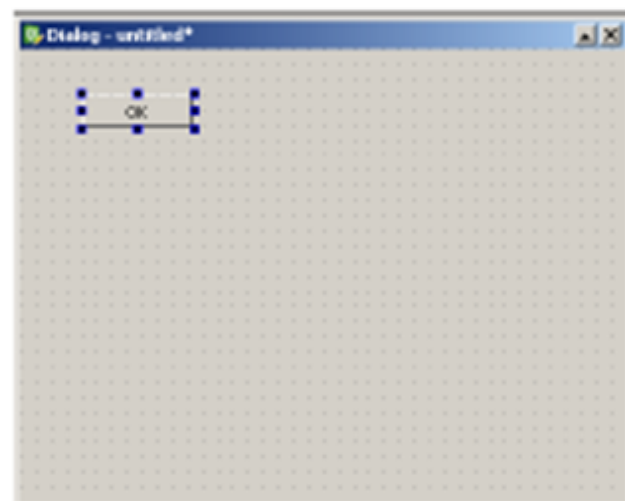
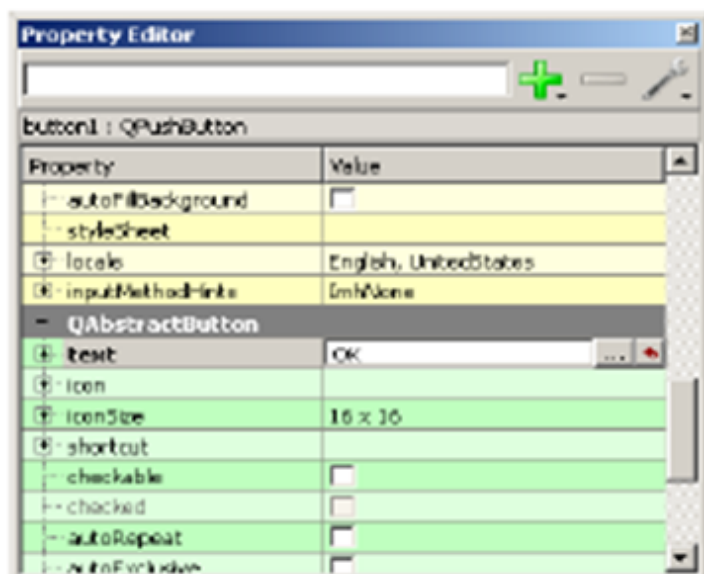
3

Using QtDesigner

The PyQt installer comes with a GUI builder tool called Qt Designer. Using its simple drag and drop interface, a GUI interface can be quickly built without having to write the code. It is however, not an IDE such as Visual Studio. Hence, Qt Designer does not have the facility to debug and build the application. Creation of a GUI interface using Qt Designer starts with choosing a top level window for the application.



You can then drag and drop required widgets from the widget box on the left pane. You can also assign value to properties of widget laid on the form.



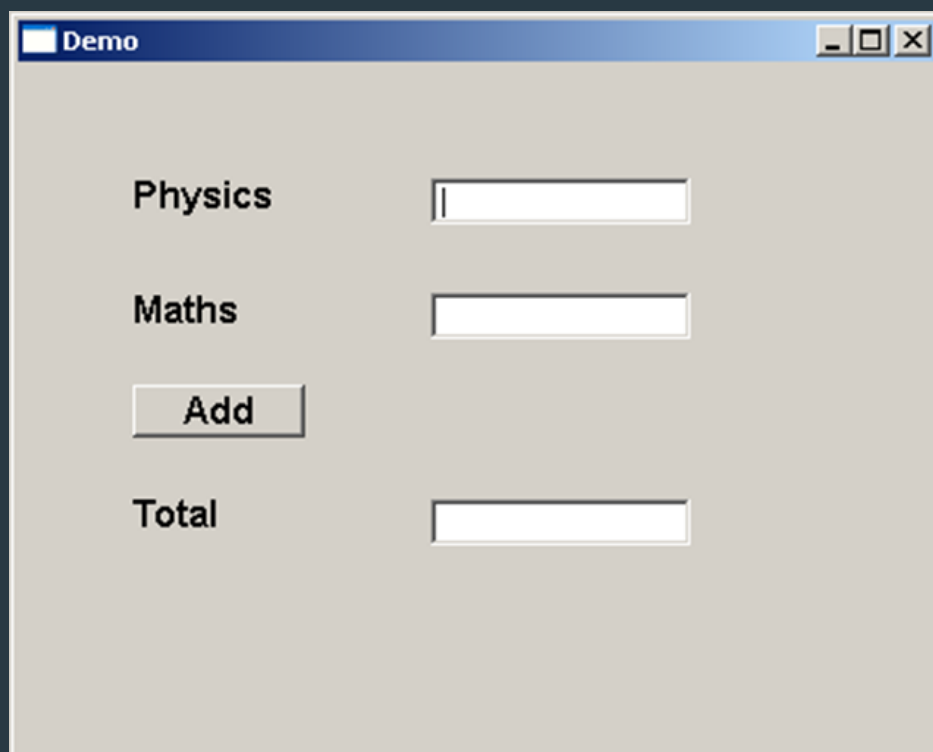
The designed form is saved as demo.ui. This ui file contains XML representation of widgets and their properties in the design. This design is translated into Python equivalent by using pyuic4 command line utility. This utility is a wrapper for uic module. The usage of pyuic4 is as follows:

```
pyuic4 -x demo.ui -o demo.py
```

In the above command, -x switch adds a small amount of additional code to the generated XML so that it becomes a self-executable standalone application.

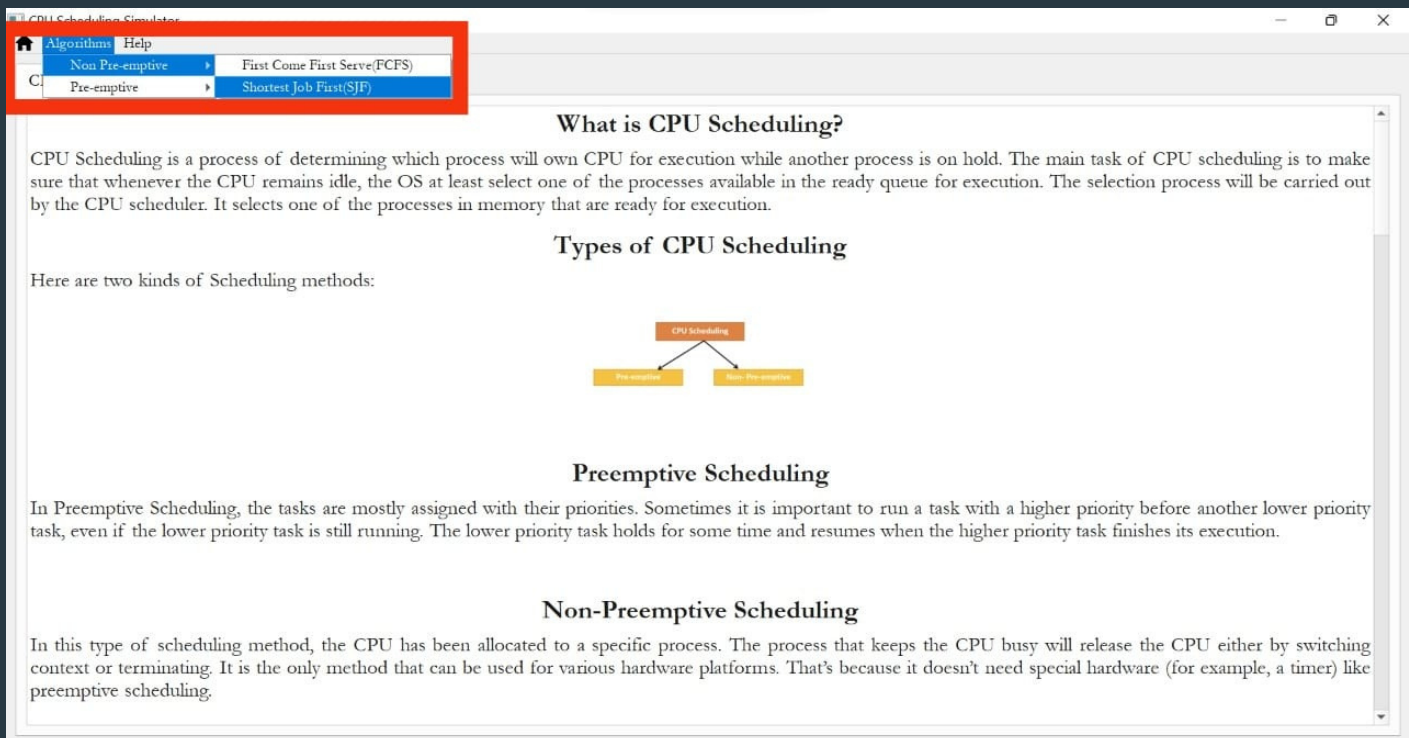
```
if __name__ == "__main__":  
    import sys  
    app = QtGui.QApplication(sys.argv)  
    Dialog = QtGui.QDialog()  
    ui = Ui_Dialog()  
    ui.setupUi(Dialog)  
    Dialog.show()  
    sys.exit(app.exec_())
```

The resultant python script is executed to show the following dialog box:

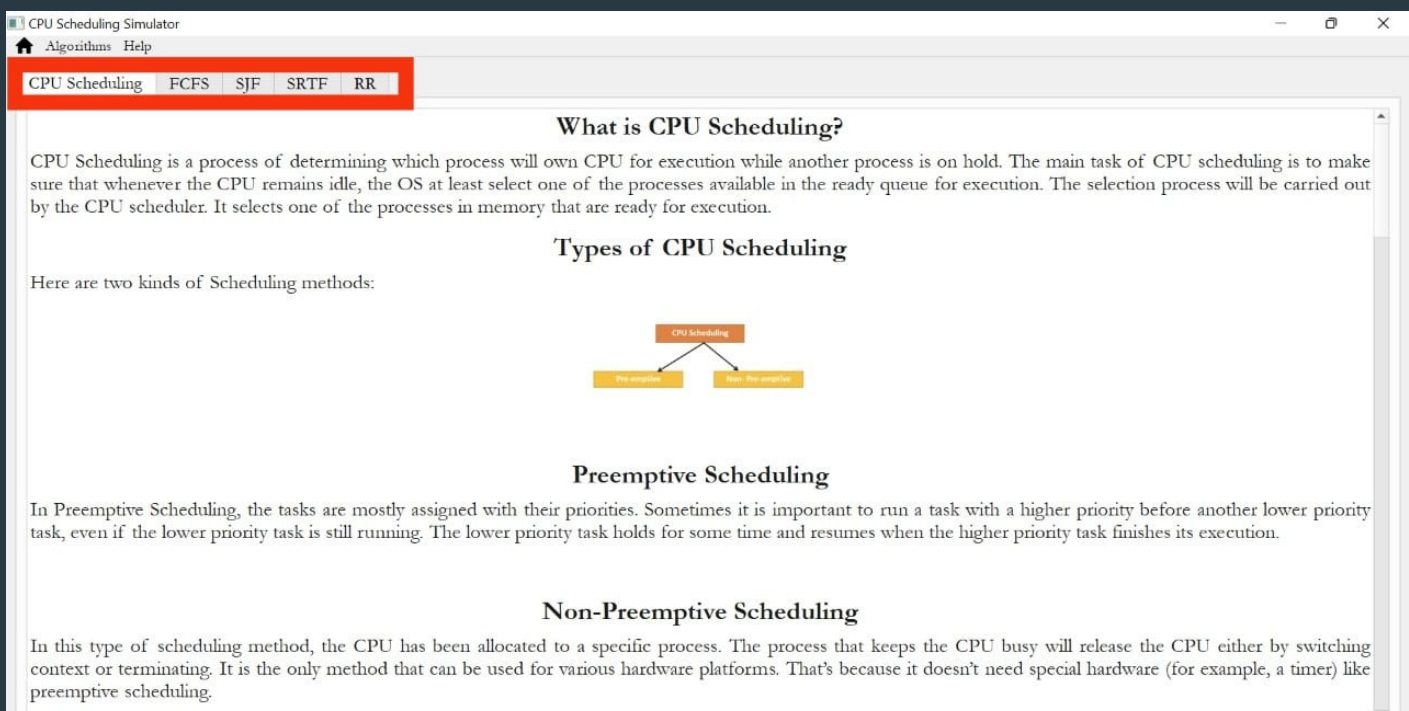


The user can input data in input fields but clicking on Add button will not generate any action as it is not associated with any function. Reacting to user-generated response is called as event handling.

Using the GUI Simulator



ALGORITHMS SIMULATOR



TABS TO GET SOME INFORMATION ABOUT TYPES OF CPU SCHEDULING ALGORITHMS

CPU Scheduling Simulator

Algorithms Help

CPU Scheduling FCFS SJF SRTF RR

What is First Come First Serve Method?

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which request the CPU first get the CPU allocation. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve. As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue and, when the CPU becomes free, it should be assigned to the process at the beginning of the queue.

Characteristics of FCFS method

- It supports non-preemptive and pre-emptive scheduling algorithms.
- Jobs are always executed on a first-come, first-serve basis.
- It is easy to implement and use.
- This method is poor in performance, and the general wait time is quite high.

Example of FCFS scheduling

A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works similarly.

Advantages of FCFS

Here, are the pros/benefits of using the FCFS scheduling algorithm:

Simulate First Come First Serve (FCFS) Algorithm

YOU CAN SIMULATE FCFS FROM HERE

CPU Scheduling Simulator

Algorithms Help

CPU Scheduling FCFS SJF SRTF RR

What is Shortest Job First Scheduling?

Shortest Job First (SJF) is an algorithm in which the process having the smallest execution time is chosen for the next execution. This scheduling method can be preemptive or non-preemptive. It significantly reduces the average waiting time for other processes awaiting execution. The full form of SJF is Shortest Job First.

There are two types of SJF methods:

- Non-Preemptive SJF
- Preemptive SJF

Characteristics of SJF Scheduling

- It is associated with each job as a unit of time to complete.
- This algorithm method is helpful for batch-type processing, where we're waiting for jobs to be complete is not critical.
- It can improve process throughput by making sure that shorter jobs are executed first, hence possibly having to have a short turnaround time.
- It improves job output by offering shorter jobs, which should be executed first, which mostly have a shorter turnaround time.

Non-Preemptive SJF

In non-preemptive scheduling, once the CPU cycle is allocated to process, the process holds it till it reaches a waiting state or is terminated.

Consider the following five processes each having its own unique burst time and arrival time.

Process Queue	Burst time	Arrival time
P1	6	2
P2	2	5

Simulate Shortest Job First (SJF) Algorithm

YOU CAN SIMULATE SJF FROM HERE

CPU Scheduling Simulator

Algorithms Help

CPU Scheduling FCFS SJF SRTF RR

What is Shortest Remaining Time First (SRTF) Scheduling Algorithm?

This Algorithm is the preemptive version of SJF scheduling. In SRTF, the execution of the process can be stopped after a certain amount of time. At the arrival of every process, the short term scheduler schedules the process with the least remaining burst time among the list of available processes and the running process.

Once all the processes are available in the ready queue, No preemption will be done and the algorithm will work as SJF scheduling. The context of the process is saved in the Process Control Block when the process is removed from the execution and the next process is scheduled. This PCB is accessed on the next execution of this process.

Example

In this example, there are five jobs P1, P2, P3, P4, P5 and P6. Their arrival time and burst time are given below in the table.

Process ID	Arrival Time	Burst Time	Completion Time	Turn Around Time	Waiting Time	Response Time
1	0	8	20	20	12	0
2	1	4	10	9	5	1
3	2	2	4	2	0	2
4	3	1	5	2	1	4
5	4	3	13	9	6	10
6	5	2	7	2	0	5

Avg Waiting Time = 24/6

The Gantt chart is prepared according to the arrival and burst time given in the table.

1. Since, at time 0, the only available process is P1 with CPU burst time 8. This is the only available process on the list therefore it is scheduled.

Simulate Shortest Remaining Time First (SRTF) Algorithm

YOU CAN SIMULATE SRTF FROM HERE

CPU Scheduling Simulator

Algorithms Help

CPU Scheduling FCFS SJF SRTF RR

What is Round-Robin Scheduling?

The name of this algorithm comes from the round-robin principle, where each person gets an equal share of something in turns. It is the oldest, simplest scheduling algorithm, which is mostly used for multitasking.

In Round-robin scheduling, each ready task runs turn by turn only in a cyclic queue for a limited time slice. This algorithm also offers starvation free execution of processes.

Characteristics of Round-Robin Scheduling

Here are the important characteristics of Round-Robin Scheduling:

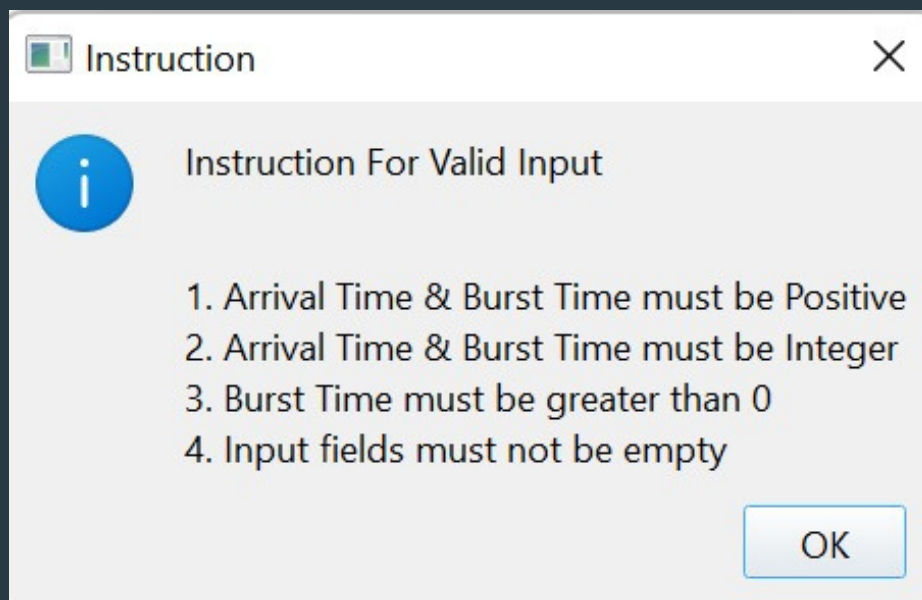
- Round robin is a pre-emptive algorithm
- The CPU is shifted to the next process after fixed interval time, which is called time quantum/time slice.
- The process that is preempted is added to the end of the queue.
- Round robin is a hybrid model which is clock-driven
- Time slice should be minimum, which is assigned for a specific task that needs to be processed. However, it may differ from OS to OS.
- It is a real-time algorithm that responds to the event within a specific time limit.
- Round robin is one of the oldest, fairest, and easiest algorithms.
- Widely used scheduling method in traditional OS.

Example of Round-Robin Scheduling

Consider the following three processes :

Simulate Round Robin (RR) Algorithm

YOU CAN SIMULATE RR FROM HERE

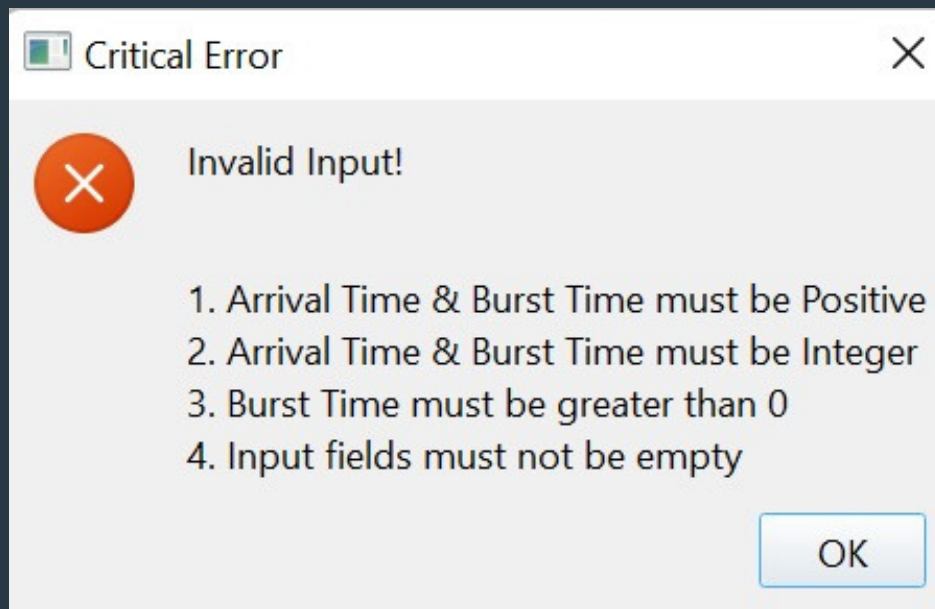


WHEN WE GO FOR SIMULATE SOME INSTRUCTIONS ABOUT VALID INPUT

The 'FCFS Simulator' window has two tabs: 'Input' and 'Output'. Under the 'Input' tab, there are two text input fields labeled 'Arrival Time' and 'Burst Time'. Below these are four buttons: 'Add Process', 'Delete Previous Process', 'Reset', and 'Execute'. At the bottom is a table with three columns: 'Process Id', 'Arrival Time', and 'Burst Time'. The table is currently empty.

Process Id	Arrival Time	Burst Time
------------	--------------	------------

SIMULATOR LIKE THIS



IF YOU PUT INVALID INPUT YOU WILL GET THIS TYPE ERROR

FCFS Simulator

Input Output

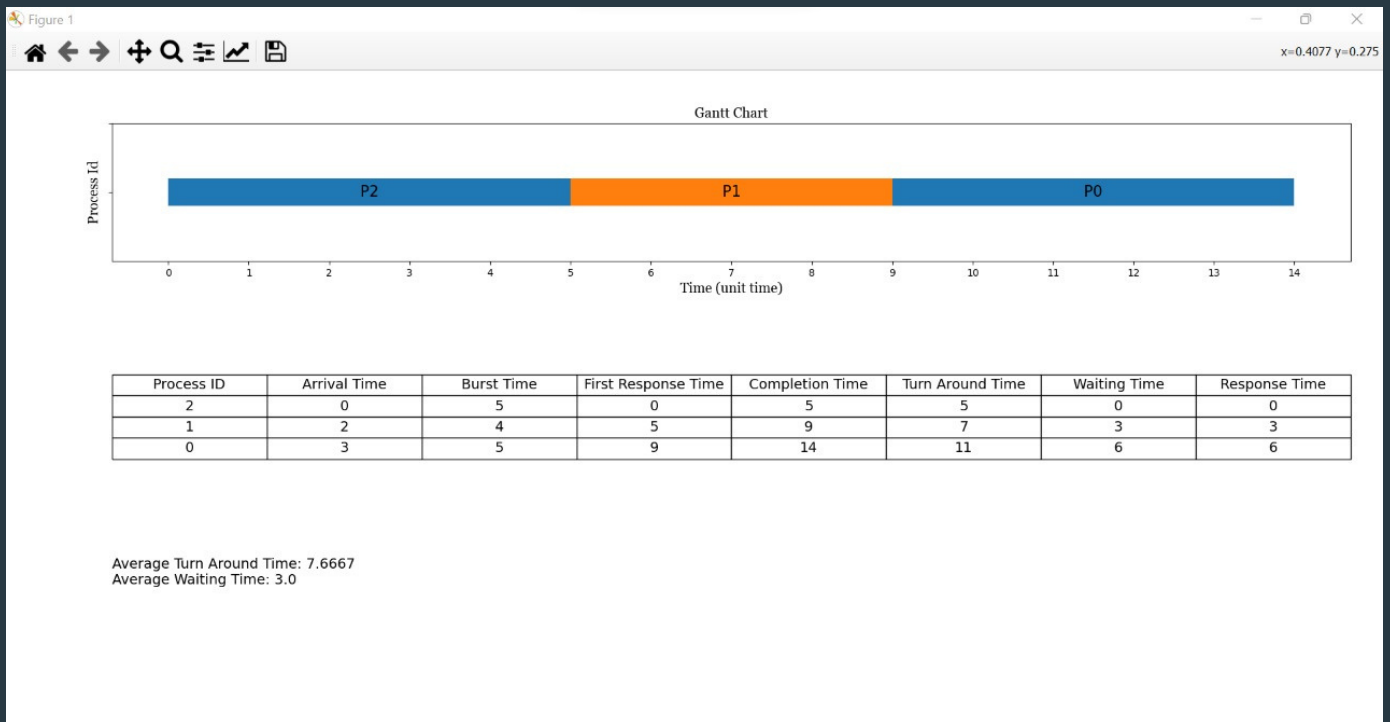
Process Id	Arrival Time	Burst Time	Completion Time (CT)	Turn Around Time (TAT)	Waiting Time (WT)
2	0	5	5	5	0
1	2	4	9	7	3
0	3	5	14	11	6

Average Waiting Time 3.0

Average Turn Around Time 7.6667

Save Result to Device Show Gantt Chart Email Me Result

OUTPUT



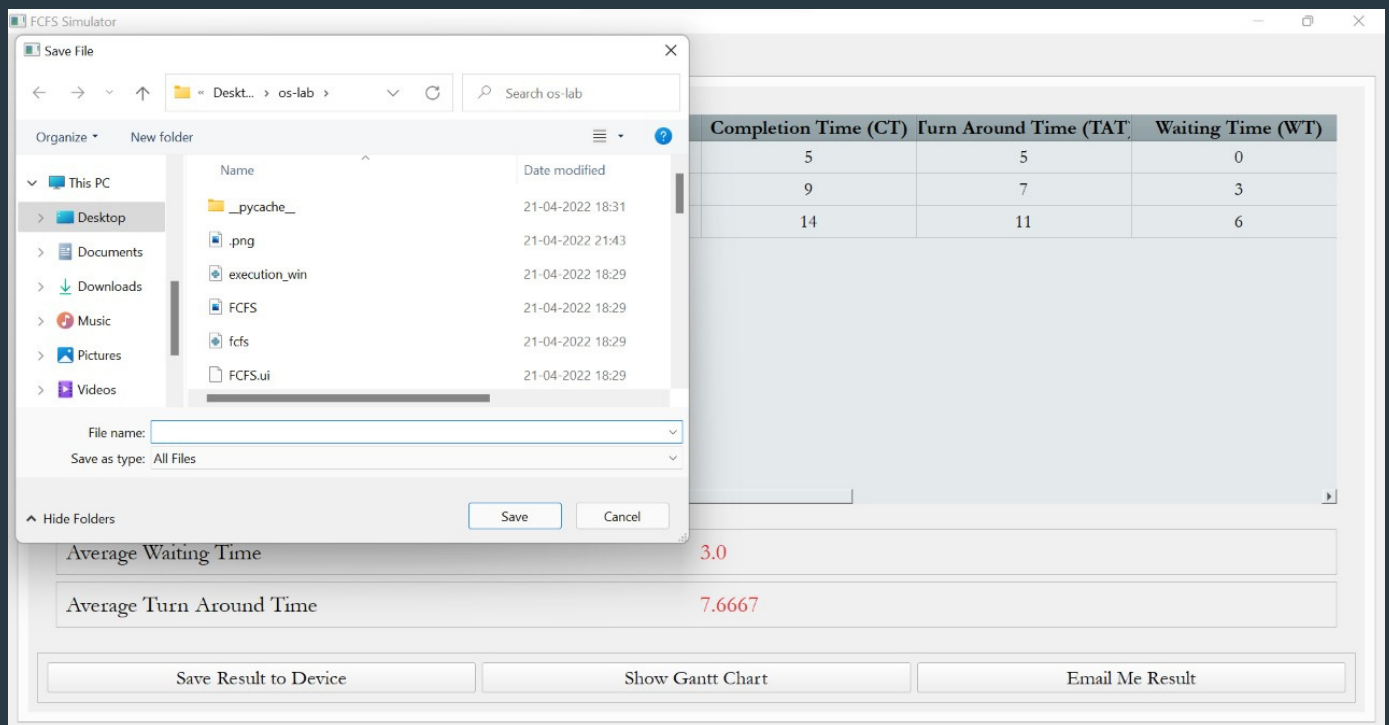
GANTT CHART OF ALGORITHM

Email
?
✕

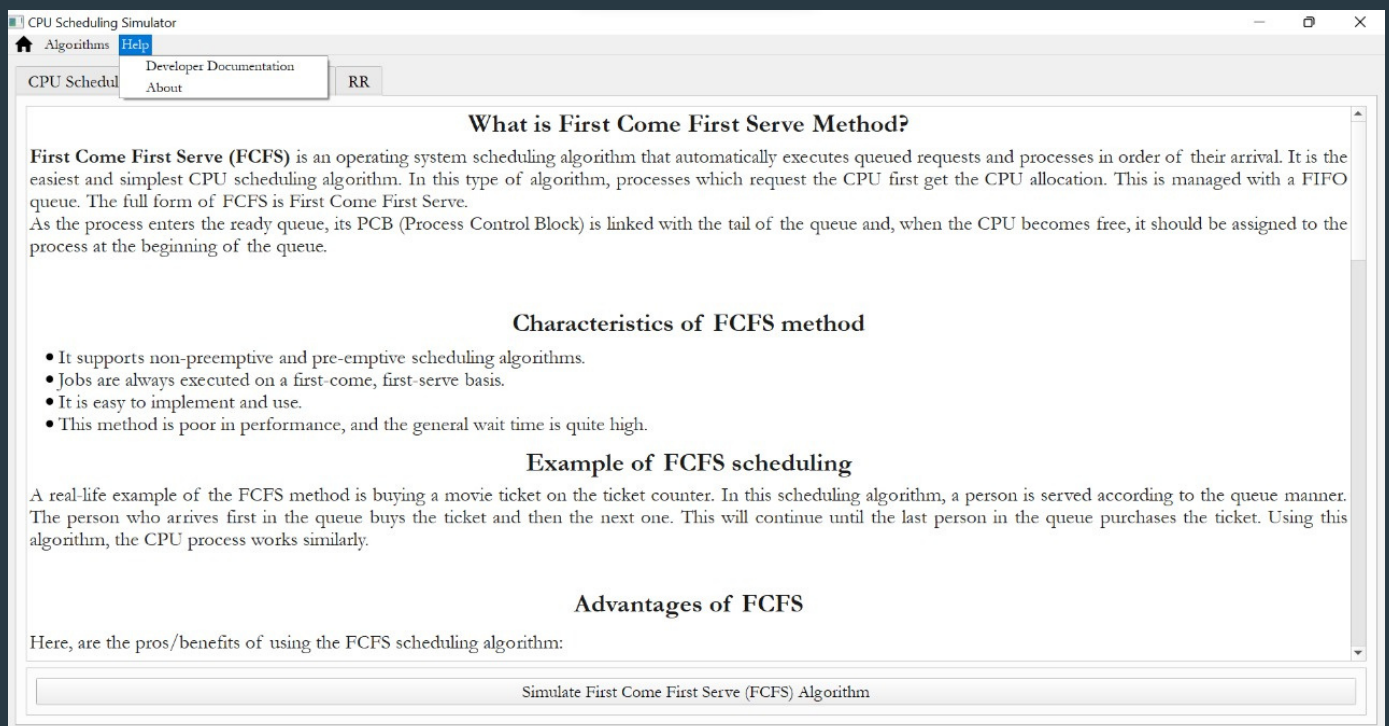
Enter Your Email:

OK
Cancel

IF YOU CLICK ON EMAIL ME RESULT



IF YOU CLICK ON SAVE RESULT TO DEVICE



IF YOU NEED SOME HELP FROM DOCUMENTS OR CONTACT WITH DEVELOPERS

Thank You

Contact Us

Hardik Inani

hardik.ice20@sot.pdpu.ac.in

Nilay Patel

nilay.pce20@sot.pdpu.ac.in

Rahul Gulati

rahul.gce20@sot.pdpu.ac.in

Dhvanil Bhagat

dhvanil.bce20@sot.pdpu.ac.in

Shrey Makadiya

shrey.mce20@sot.pdpu.ac.in

Drashti Bhavsar

drashti.bce20@sot.pdpu.ac.in