Министерство науки и высшего образования
Российской Федерации

Федеральное государственное бюджетное
образовательное учреждения высшего образования

“НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ”

Кафедра теоретической и прикладной информатики

Расчетно-графическое задание
по дисциплине “Операционные системы, среды и оболочки”

«Разработка эмулятора диспетчера ОС.

Круговой циклический алгоритм»

Факультет: ПМИ

Группа: ПМИ-02

Бригада: 2

Студент: Сидоров Даниил

Преподаватель: Кобылянский Валерий Григорьевич

Новосибирск
2022

Оглавление

Введение

Текст задания

Разработать программу – эмулятор диспетчера ОС на основе кругового циклического алгоритма.

Планирование процессов включает в себя решение следующих задач:

· определение момента времени для смены выполняемого процесса;

· выбор процесса на выполнение из очереди готовых процессов.

Различные алгоритмы планирования могут преследовать различные цели и обеспечивать разное качество мультипрограммирования. Например, алгоритм должен гарантировать, что ни один процесс не будет занимать процессор дольше определенного времени; другой обеспечивает максимально быстрое выполнение «коротких» задач; третий обеспечивает преимущественное право на процессорное время интерактивным приложениям. Именно особенности планирования процессов в наибольшей степени определяют специфику ОС.

В большинстве ОС универсального назначения планирование осуществляется динамически, то есть решения принимаются во время работы системы на основе анализа текущей ситуации. ОС не имеет никакой предварительной информации о задачах, которые появляются в случайные моменты времени.

Статический тип планирования используется в специализированных системах, где набор одновременно выполняемых задач определен заранее (например, в системах реального времени). Здесь решение о планировании принимается заранее.

Диспетчеризация заключается в реализации найденного в результате планирования решения, т. е. в переключении процессора с одного потока на другой, и сводится к следующему:

· сохранение контекста текущего процесса;

· загрузка контекста нового процесса;

· запуск нового процесса.

В отличие от планирования, осуществляемого программными средствами ОС, диспетчеризация реализуется совместно с аппаратными средствами процессора. В различных ОС компоненты, занимающиеся планированием, могут называться по-разному: scheduler – распорядитель, или планировщик, – в Unix; dispatcher – в Windows.

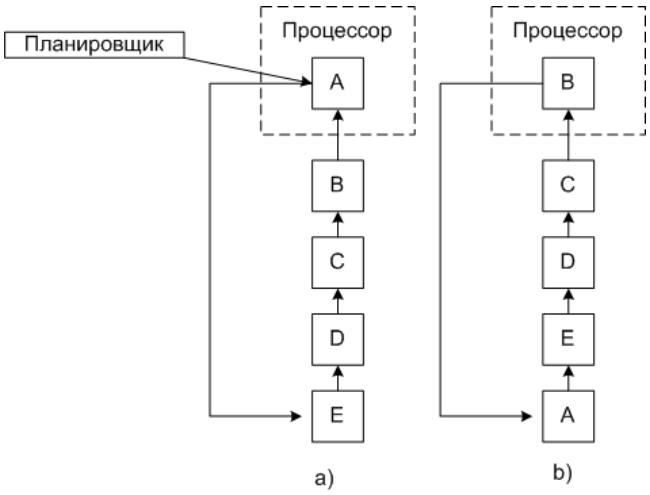
Существует шесть популярных алгоритмов планирования процессов:

- Планирование «первым пришел – первым обслужен» (FCFS)
- Планирование Shortest-Job-Next (SJN)
- Приоритетное планирование
- Самое короткое оставшееся время
- Круговой циклический алгоритм
- Планирование многоуровневых очередей

Эти алгоритмы являются либо не вытесняющими, либо вытесняющими. Непрерывающие алгоритмы разработаны таким образом, что, как только процесс входит в рабочее состояние, он не может быть прерван до тех пор, пока не завершит свое выделенное время, тогда как упреждающее планирование основано на приоритете, когда планировщик может выгрузить процесс с низким приоритетом в любое время, когда процесс с высоким приоритетом переходит в состояние готовности.

Теоретическая часть

Круговой циклический алгоритм - алгоритм планирования вытесняющих процессов. Каждому процессу предоставляется определенное время для выполнения, оно называется **квантом**. Как только процесс выполняется в течение заданного периода времени, он прерывается, и другой процесс выполняется в течение заданного периода времени. Название алгоритма происходит от принципа циклического перебора, известного из других областей, когда каждый человек по очереди получает равную долю чего-либо. Это самый простой алгоритм планирования и часто используемый.



Преимущества:

- Простота
- Справедливость (каждому процессу одинаковое время)

Недостатки:

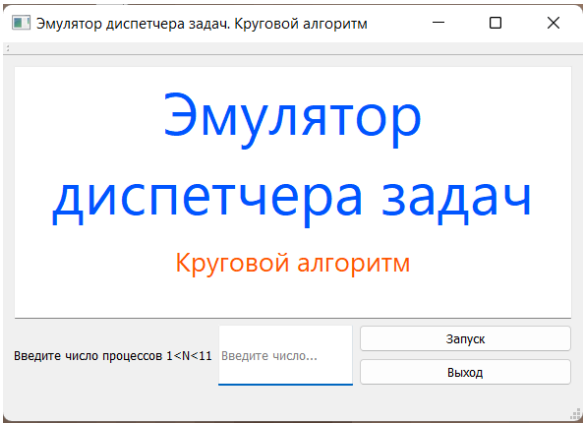
- Если частые переключения, то происходит уменьшение производительности.
- Если редкие переключения, то происходит увеличение времени ответа на запрос.
- Отсутствие приоритетности.

Описание программы

Программа была выполнена с помощью платформы Qt, которая позволяет создавать приложения с помощью графического пользовательского интерфейса. Используемый язык программирования – C++.

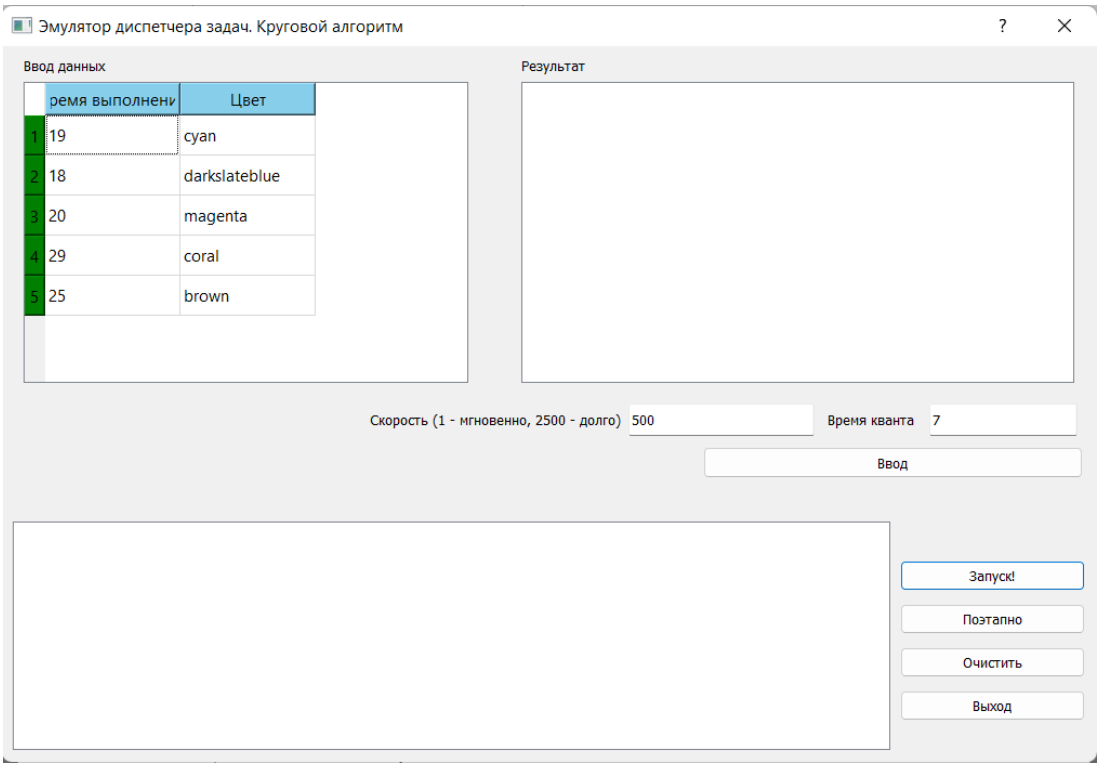
Приложение содержит в себе 2 формы:

1) Первая форма, в которой происходит ввод количества процессов:



Форма содержит в себе название программы, текстовый блок, в котором вводится число процессов, кнопку ”Выход”, которая позволяет завершить программу, и кнопку “Запуск”, которая открывает вторую форму.

2) Вторая форма является основной. В ней происходит основная работа программы.



Форма содержит 2 таблицы: таблица для ввода данных и таблица результатов, которая появляется после выполнения процессов со всеми выходными данными.

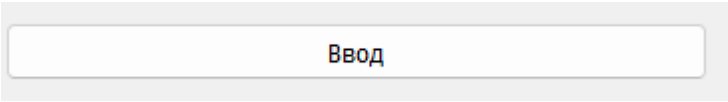
	Время выполнения	Цвет
1	19	cyan
2	18	darkslateblue
3	20	magenta
4	29	coral
5	25	brown

	ID	Время окончания	Время ожидания
1	1	75	56
2	2	79	61
3	3	85	65
4	4	111	82
5	5	110	85

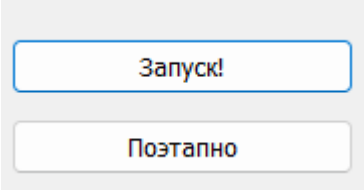
При открытии основной формы, таблица ввода данных заполняется случайными величинами для ускорения тестирования.

Форма так же имеет 2 текстовых поля: поле для ввода скорости выполнения программы и поле для ввода кванта времени.

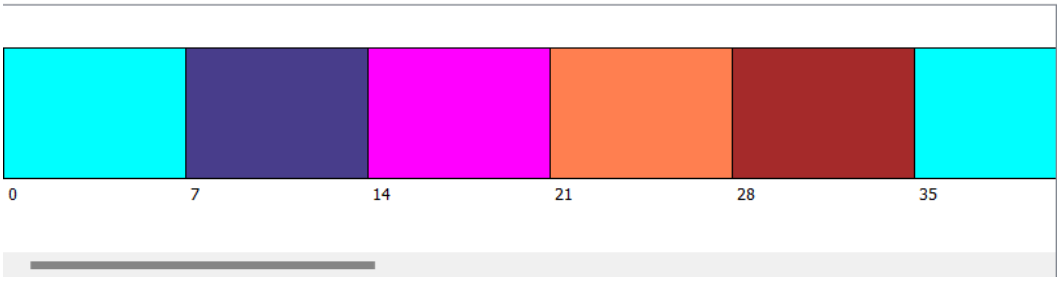
Чтобы программа приняла входные данные, необходимо нажать кнопку “Ввод”.



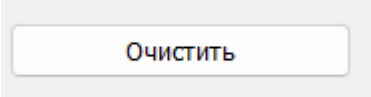
Программа имеет два режима работы: автоматический и поэтапный. Чтобы выбрать режим существуют две кнопки: “Запуск!” и “Поэтапно”.



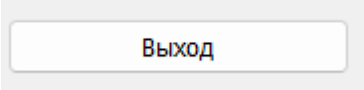
В графической области происходит зарисовка в виде прямоугольников тех процессов, которые выполняются в данный квант.



Существует кнопка “Очистить” для очищения графическом области. Её необходимо нажать для корректного повторного запуска программы.



Кнопка “Выход” возвращает к первой форме.



Список функций:

Обработка событий:

void on_pushButton_clicked(): Событие, возникающее при нажатии кнопки “Ввод”. Программа получает данные из таблицы ввода данных, а также квант времени и скорость.

void on_pushButton_2_clicked():Событие, возникающее при нажатии кнопки “Запуск”. Программа проверит наличие входных данных и запустит круговой циклический алгоритм в автоматическом режиме.

void on_pushButton_3_clicked():Событие, возникающее при нажатии кнопки “Выход”. Вторая форма закроется и управление перейдет к первой форме.

void on_pushButton_4_clicked():Событие, возникающее при нажатии кнопки “Поэтапно”. Программа проверит наличие входных данных и запустит круговой циклический алгоритм в поэтапной режиме.

void on_pushButton_5_clicked():Событие, возникающее при нажатии кнопки “Очистить”. Программа очистит графическую область.

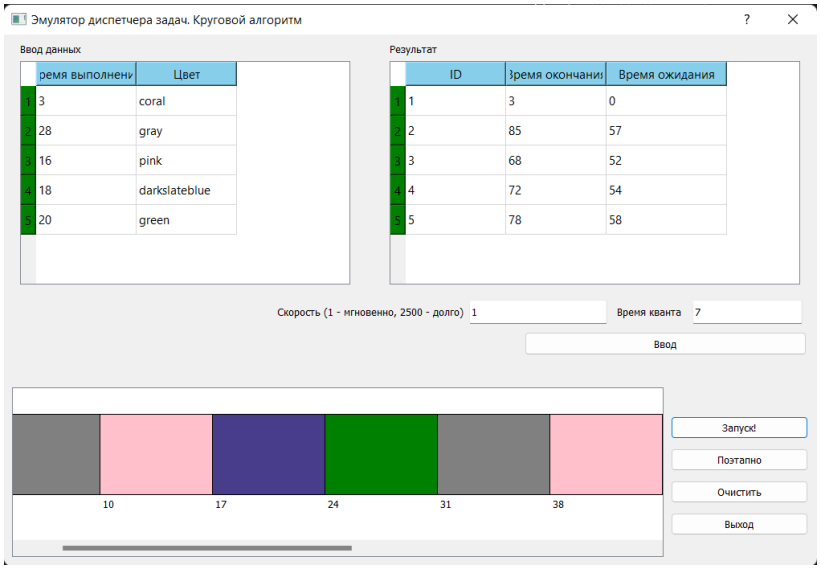
Подпрограммы:

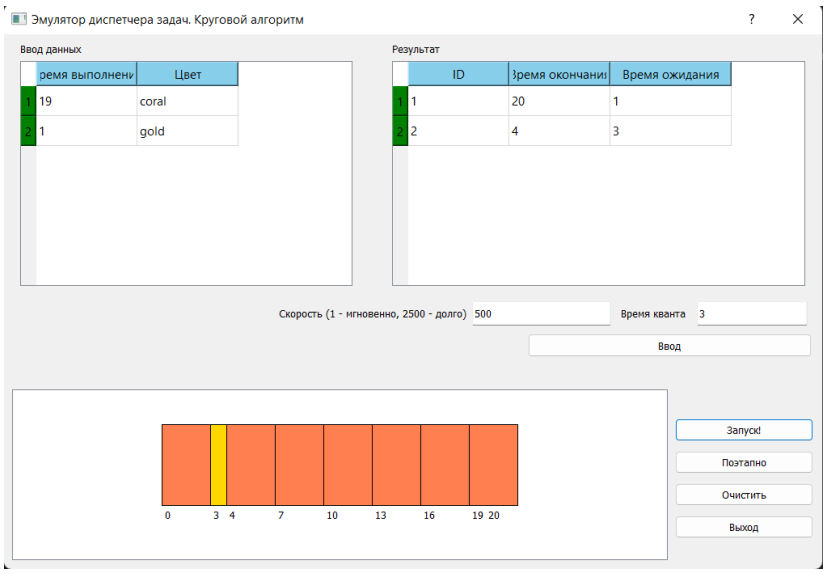
void displayturnwait(): Программа, формирующая таблицу результатов и заполняющая ее выходными данными.

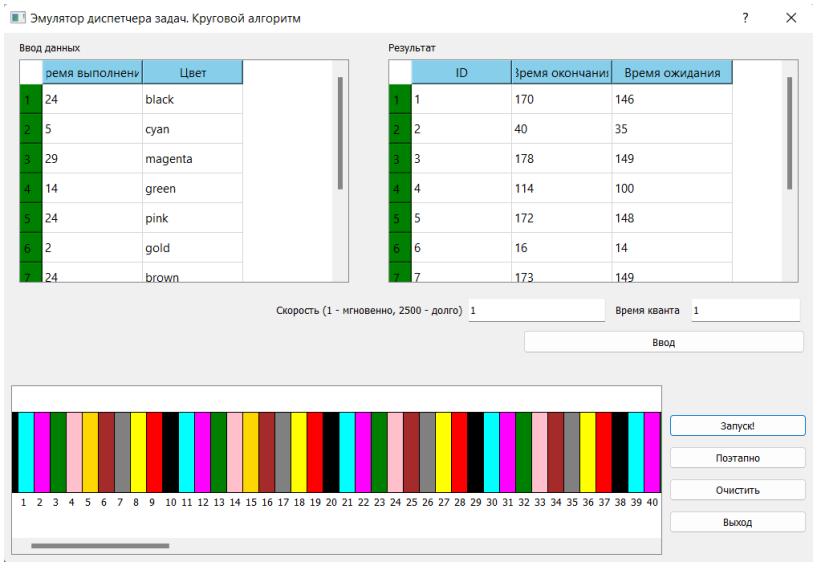
Результаты тестирования

Во всех тестах получили ожидаемо корректные результаты.

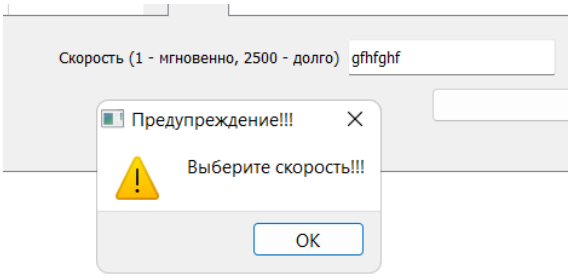
Сделаем несколько тестов с разными входными данными:



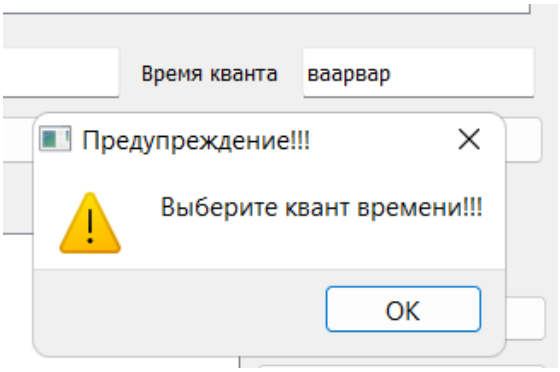




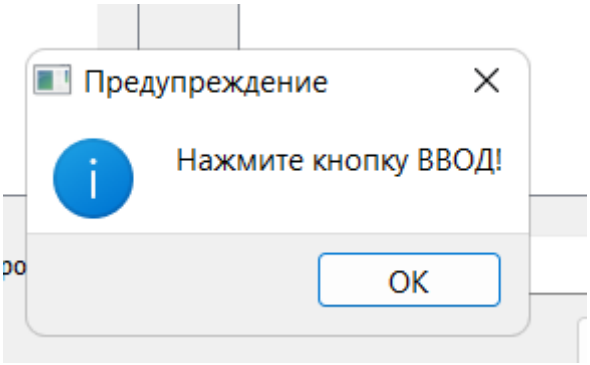
При попытке ввести неверную скорость появляется предупреждение:



При попытке ввести неверный квант времени появляется предупреждение:

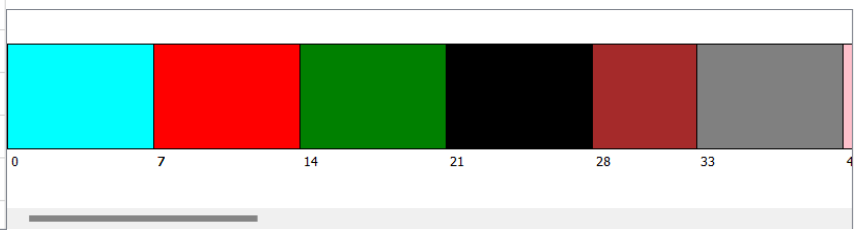


При попытке запуска программы в автоматическом или поэтапном режиме без нажатия кнопки “Ввод” появляется предупреждение:



При некорректном вводе времени выполнения или цвета процесс не будет выполнен:

	Время выполнения	Цвет
1	8	cyan
2	fdgd	coral
3	gfdg	yellow
4	29	red
5	19	green
6	29	black
7	5	brown



Заключение

Мы разработали программу – эмулятор диспетчера ОС с круговым циклическим алгоритмом.

Список использованных источников

Алгоритмы планирования операционной системы

Алгоритмы планирования процессов и потоков. Алиса Королева

Циклический алгоритм – Википедия (Wikipedia.org)

Операционные системы. Лекции. Богомолов В.А.