

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО «СЕВЕРО - КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных технологий
Кафедра инфокоммуникаций.

Курсовая работа

По дисциплине
«Операционные системы»
на тему:

«Разработка подсистемы управления процессами с вытесняющей
многозадачностью»

Выполнил:

Абдуллаев Джамалудин
Магомедзагирович
Студент 2 курса группы ИВТ-б-о-19-
1 специальности «Информатика и
вычислительная техника» очной
формы обучения

(подпись)

Руководитель работы:

Гайчук Дмитрий Викторович
канд. техн. наук,
доцент кафедры ПМБК

Работа допущена к защите _____
(подпись руководителя) (дата)

Работа выполнена и
защищена с оценкой _____ Дата защиты _____

Член комиссии: _____
(должность) (подпись) (И.О. Фамилия)

_____	_____	_____
_____	_____	_____
_____	_____	_____

Ставрополь, 2021 г.

Содержание

Введение	3
1. Разработка подсистемы управления процессами с вытесняющей многозадачностью	3
1.2 Понятие подсистемы управления процессами.....	3
1.3 Понятие вытесняющей многозадачности	4
1.4 Постановка задачи	5
2. Описание программы	6
2.1 Функциональное назначение	6
2.2 Технические средства, используемые при создании программы	6
2.3 Логическая структура программы.....	8
2.4 Входные и выходные данные	10
3. Руководство оператора	10
3.1 Общие сведения и назначение программы	10
3.2 Графический интерфейс	10
3.3 Работа с программой.....	11
3.4 Основные характеристики программы	14
4. Программа и методика испытаний.....	15
4.1 Объект испытаний.....	15
4.2 Цель испытаний.....	15
4.3 Требования к программе	15
4.4 Методы испытаний	16
Заключение	16
Библиография	16
Исходный код программы	17

Введение

Важнейшей частью операционной системы, непосредственно влияющей на функционирование вычислительной машины, является подсистема управления процессами. Для каждого вновь создаваемого процесса ОС генерирует системные информационные структуры, которые содержат данные о потребностях процесса в ресурсах вычислительной системы, а также о фактически выделенных ему ресурсах. Таким образом, процесс можно также определить как некоторую заявку на потребление системных ресурсов.

Чтобы процесс мог быть выполнен, операционная система должна назначить ему область оперативной памяти, в которой будут размещены коды и данные процесса, а также предоставить ему необходимое количество процессорного времени. Кроме того, процессу может понадобиться доступ к таким ресурсам, как файлы и устройства ввода-вывода.

В данной курсовой работе необходимо реализовать подсистему управления процессами с вытесняющей многозадачностью

1. Разработка подсистемы управления процессами с вытесняющей многозадачностью

1.2 Понятие подсистемы управления процессами

Приоритетное обслуживание предполагает наличие у потоков некоторой изначально известной характеристики — приоритета, на основании которой определяется порядок их выполнения. Приоритет — это число, характеризующее степень привилегированности потока при использовании ресурсов вычислительной машины, в частности процессорного времени: чем выше приоритет, тем выше привилегии, тем меньше времени будет проводить поток в очередях.

Приоритет процесса назначается операционной системой при его создании. Значение приоритета включается в описатель процесса и используется при назначении приоритета потокам этого процесса. При назначении приоритета вновь созданному процессу ОС учитывает, является этот процесс системным или прикладным, каков статус пользователя, запустившего процесс, было ли явное указание пользователя на присвоение процессу определенного уровня приоритета. Поток может быть инициирован не только по команде пользователя, но и в результате выполнения системного вызова другим потоком.

Изменение приоритета могут происходить по инициативе самого потока, когда он обращается с соответствующим вызовом к операционной системе, или по инициативе пользователя, когда он выполняет соответствующую команду.

1.3 Понятие вытесняющей многозадачности

Существует два основных типа процедур планирования процессов - вытесняющие (preemptive) и невытесняющие (non-preemptive).

Вытесняющая многозадачность — это такой способ, при котором решение о переключении процессора с выполнения одного процесса на выполнение другого процесса принимается планировщиком операционной системы, а не самой активной задачей.

Понятия preemptive и non-preemptive иногда отождествляются с понятиями приоритетных и беспriorитетных дисциплин, что совершенно неверно, а также с понятиями абсолютных и относительных приоритетов, что неверно отчасти. Вытесняющая и невытесняющая многозадачность — это более широкие понятия, чем типы приоритетности. Приоритеты задач могут как использоваться, так и не использоваться и при вытесняющих, и при невытесняющих способах планирования. Так в случае использования приоритетов дисциплина относительных приоритетов может быть отнесена к классу систем

с невытесняющей многозадачностью, а дисциплина абсолютных приоритетов - к классу систем с вытесняющей многозадачностью. А беспriorитетная дисциплина планирования, основанная на выделении равных квантов времени для всех задач, относится к вытесняющим алгоритмам. Моя тема курсовой работы связана именно подсистемой управления процессами с вытесняющей многозадачностью.

1.4 Постановка задачи

При вытесняющей многозадачности механизм планирования задач целиком сосредоточен в операционной системе, и программист пишет свое приложение, не заботясь о том, что оно будет выполняться параллельно с другими задачами. При этом операционная система выполняет следующие функции: определяет момент снятия с выполнения активной задачи, запоминает ее контекст, выбирает из очереди готовых задач следующую и запускает ее на выполнение, загружая ее контекст.

Для того, чтобы обеспечить хорошее время реакции системы, алгоритм планирования использует квантование. В соответствии с этой концепцией процессы выполняются по порядку их поступления.

Так же мы используем принцип FIFO. Выталкивание первой пришедшей страницы (FIFO). При выталкивании страниц по принципу FIFO мы присваиваем каждой странице в момент поступления в основную память временную метку. Когда появляется необходимость удалить из основной памяти какую-нибудь страницу, мы выбираем ту, которая находилась в памяти дольше других. Интуитивный аргумент в пользу подобной стратегии кажется весьма весомым, а именно: у данной страницы уже были возможности “использовать свой шанс”, и пора дать подобные возможности и другой странице. К сожалению, стратегия FIFO с достаточно большой вероятностью будет приводить к замещению активно используемых страниц, поскольку тот факт, что страница находится в основной памяти в течение длительного времени, вполне может означать, что она постоянно в работе.

Например, для крупных систем разделения времени стандартна ситуация, когда многие пользователи во время ввода и отработки своих программ совместно используют одну копию текстового редактора. Если в подобной системе выталкивать страницы по принципу FIFO, это может привести к удалению из памяти какой-либо интенсивно используемой странице редактора. А это будет безусловно нецелесообразно, поскольку её почти немедленно придется снова переписывать в основную память.

2. Описание программы

2.1 Функциональное назначение

Подсистема управления процессами планирует выполнение процессов, то есть занимается созданием и уничтожением процессов, обеспечивает процессы необходимыми системными ресурсами, поддерживает взаимодействие между процессами.

Процесс (или по-другому, задача) - абстракция, описывающая выполняющуюся программу. Для операционной системы процесс представляет собой единицу работы, заявку на потребление системных ресурсов.

2.2 Технические средства, используемые при создании программы

Python поддерживает параллельное выполнение кода через многопоточность. Поток — это независимый путь исполнения, способный выполняться одновременно с другими потоками.

Программа на языке программирования Python запускается как единственный поток, автоматически создаваемый главный поток и становится многопоточной при помощи создания дополнительных потоков.

Управление многопоточностью осуществляет планировщик потоков. Планировщик потоков гарантирует, что активным потокам выделяется соответствующее время на выполнение.

На однопроцессорных компьютерах планировщик потоков использует квантование времени – быстрое переключение между выполнением каждого из активных потоков.

На многопроцессорных компьютерах многопоточность реализована как смесь квантования времени и подлинного параллелизма. Необходимость квантования времени все равно остается, так как операционная система должна обслуживать как свои собственные потоки, так и потоки других приложений.

Говорят, что поток вытесняется, когда его выполнение приостанавливается из-за внешних факторов типа квантования времени. В большинстве случаев поток не может контролировать, когда и где он будет вытеснен.

Все потоки одного приложения логически содержатся в пределах процесса – модуля операционной системы, в котором выполняется приложение.

Приоритеты потоков и процессов

Свойство приоритетов определяет, сколько времени на исполнение будет выделено потоку относительно других потоков того же процесса. Существует 6 градаций приоритета потока: Real time class, High class, Above normal class, Normal class, Below normal class, Idle class.

Значение приоритета становится существенным, когда одновременно исполняются несколько потоков.

Установка приоритета потока на максимум еще не означает работу в реальном времени (real-time), так как существуют еще приоритет процесса приложения.

Если real-time приложение имеет пользовательский интерфейс, может быть не желательно поднимать приоритет его процесса, так как обновление

экрана будет съедать чересчур много времени CPU – тормозя весь компьютер, особенно если UI достаточно сложный.

2.3 Логическая структура программы

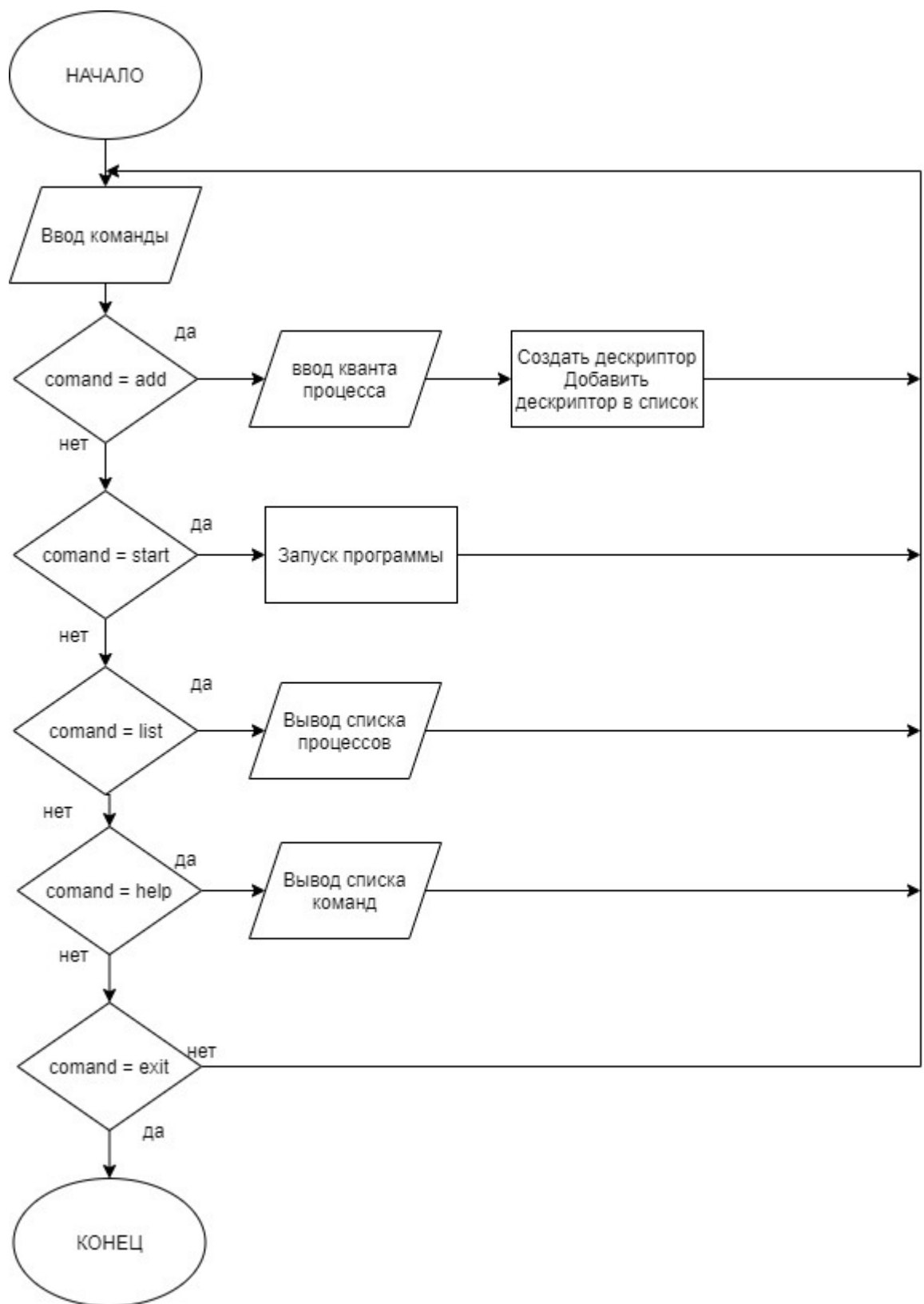


Рис. 2. Алгоритм программы

Основу программы составляет функция CreateProc. В ней реализована абстрактная модель алгоритма планирования на основе вытесняющей многозадачности. Каждому из процессов задается квант (число). Чем больше показатель кванта, тем больше итераций выполнится с процессом.

2.4 Входные и выходные данные

В качестве входных данных для нормального функционирования алгоритма планирования на основе квантования, пользователю требуется ввести квант процесса (целое число)

На выходе мы получаем работающий алгоритм планирования на основе квантования.

3. Руководство оператора

3.1 Общие сведения и назначение программы

Программа написана в среде PyCharm 2021.1.1 на языке Python и представляет собой менеджер управления процессами, построенный на современных средствах управления, с понятным интерфейсом. Программа является полностью автономной и не требует установки другого программного обеспечения. В основной части руководства будут описаны возможности программы, описание основных характеристик и особенностей программы.

3.2 Графический интерфейс

После открытия программы, пользователю предоставляется графический интерфейс.

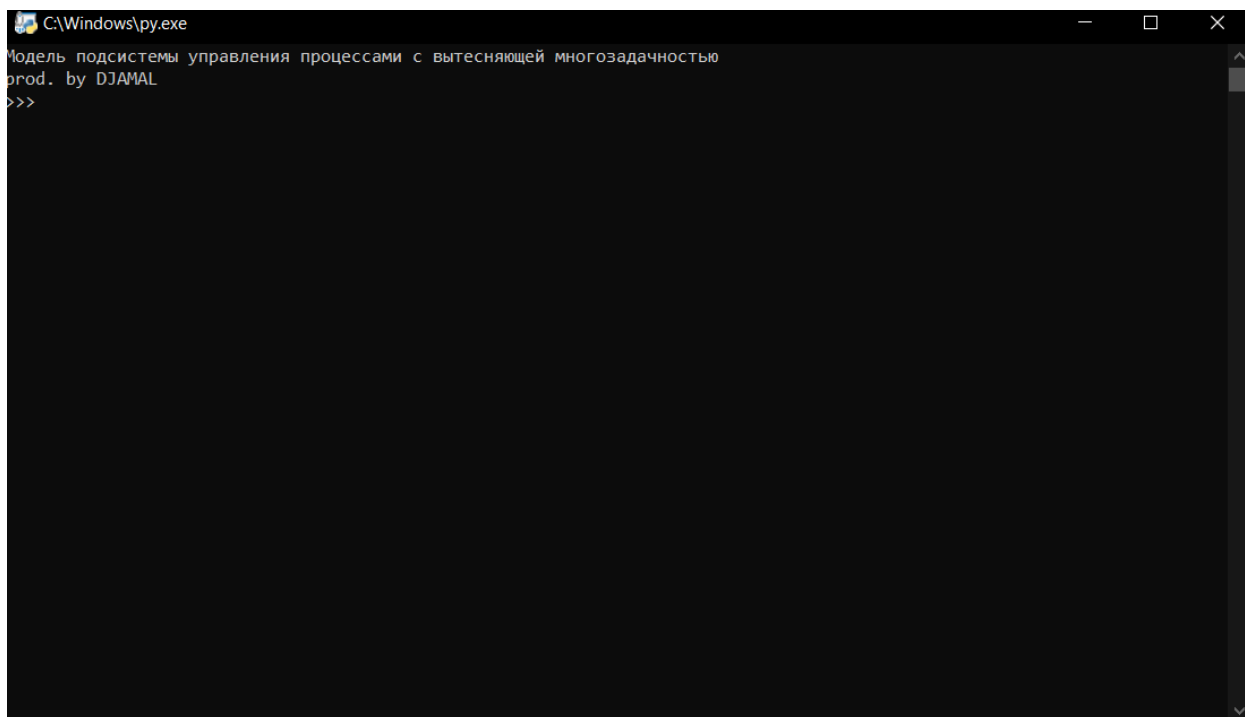


Рис. 3. Главное окно программы после запуска

Область командной строки позволяет ввести команду, которую нам необходимо отобразить.

Команда "add" – запускает создание модели процесса.

Команда "list" – выводит список процессов.

Команда "help" – выводит список команд с кратким описанием.

Команда "start" – запускает алгоритм планирования.

Команда "exit" – завершает работу программы.

3.3 Работа с программой

Запуск процессов выполняется с помощью команды “start”, однако перед этим следует создать процессы при помощи команды “add”

```

E:\Курсовая>variant3.py
Модель подсистемы управления процессами с вытесняющей многозадачностью
prod. by DJAMAL
>>> add
Введите квант: 7
>>> add
Введите квант: 5
>>> add
Введите квант: 4
>>> add
Введите квант: 8
>>>

```

Рис. 4. Добавление процессов.

```

>>> start

```

ИТЕРАЦИЯ № 1	
Выполнение процесса № 0 .	Осталось квантов: 6
Выполнение процесса № 1 .	Осталось квантов: 4
Выполнение процесса № 2 .	Осталось квантов: 3
Выполнение процесса № 3 .	Осталось квантов: 7

ИТЕРАЦИЯ № 2	
Выполнение процесса № 0 .	Осталось квантов: 5
Выполнение процесса № 1 .	Осталось квантов: 3
Выполнение процесса № 2 .	Осталось квантов: 2
Выполнение процесса № 3 .	Осталось квантов: 6

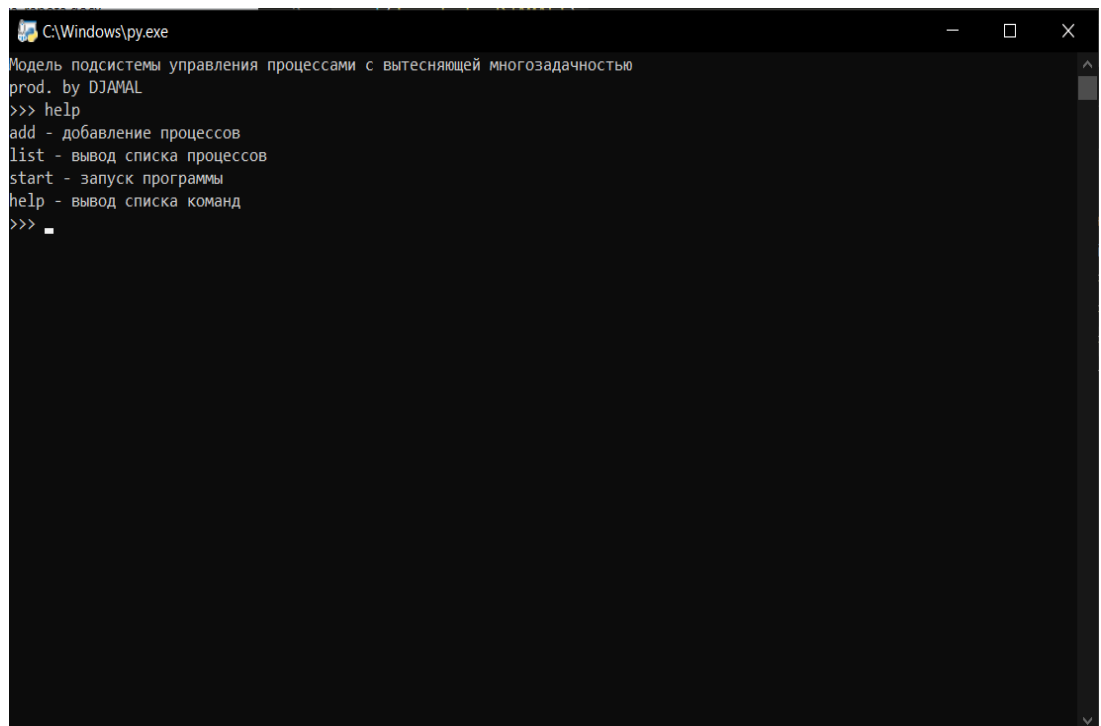
ИТЕРАЦИЯ № 3	
Выполнение процесса № 0 .	Осталось квантов: 4
Выполнение процесса № 1 .	Осталось квантов: 2
Выполнение процесса № 2 .	Осталось квантов: 1
Выполнение процесса № 3 .	Осталось квантов: 5

ИТЕРАЦИЯ № 4	
Выполнение процесса № 0 .	Осталось квантов: 3
Выполнение процесса № 1 .	Осталось квантов: 1
Выполнение процесса № 2 .	Осталось квантов: 0
Процесс № 2 завершен	
Выполнение процесса № 3 .	Осталось квантов: 4

ИТЕРАЦИЯ № 5	
Выполнение процесса № 0 .	Осталось квантов: 2
Выполнение процесса № 1 .	Осталось квантов: 0
Процесс № 1 завершен	

Рис. 5. Запуск алгоритма планирования.

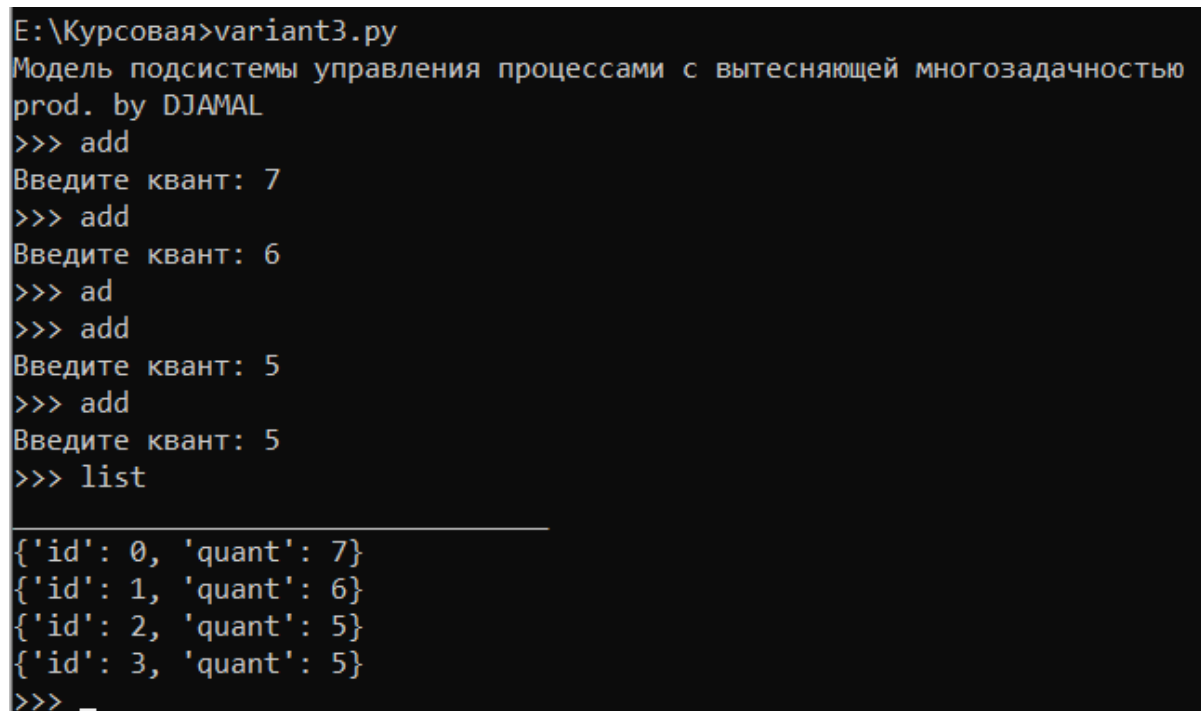
воспользоваться командой «help», для вывода списка всех команд с кратким описанием.



```
C:\Windows\py.exe
Модель подсистемы управления процессами с вытесняющей многозадачностью
prod. by DJAMAL
>>> help
add - добавление процессов
list - вывод списка процессов
start - запуск программы
help - вывод списка команд
>>> .
```

Рис. 7. Использование команды «help»

Для того чтобы отобразить список процессов, можно воспользоваться командой «list»

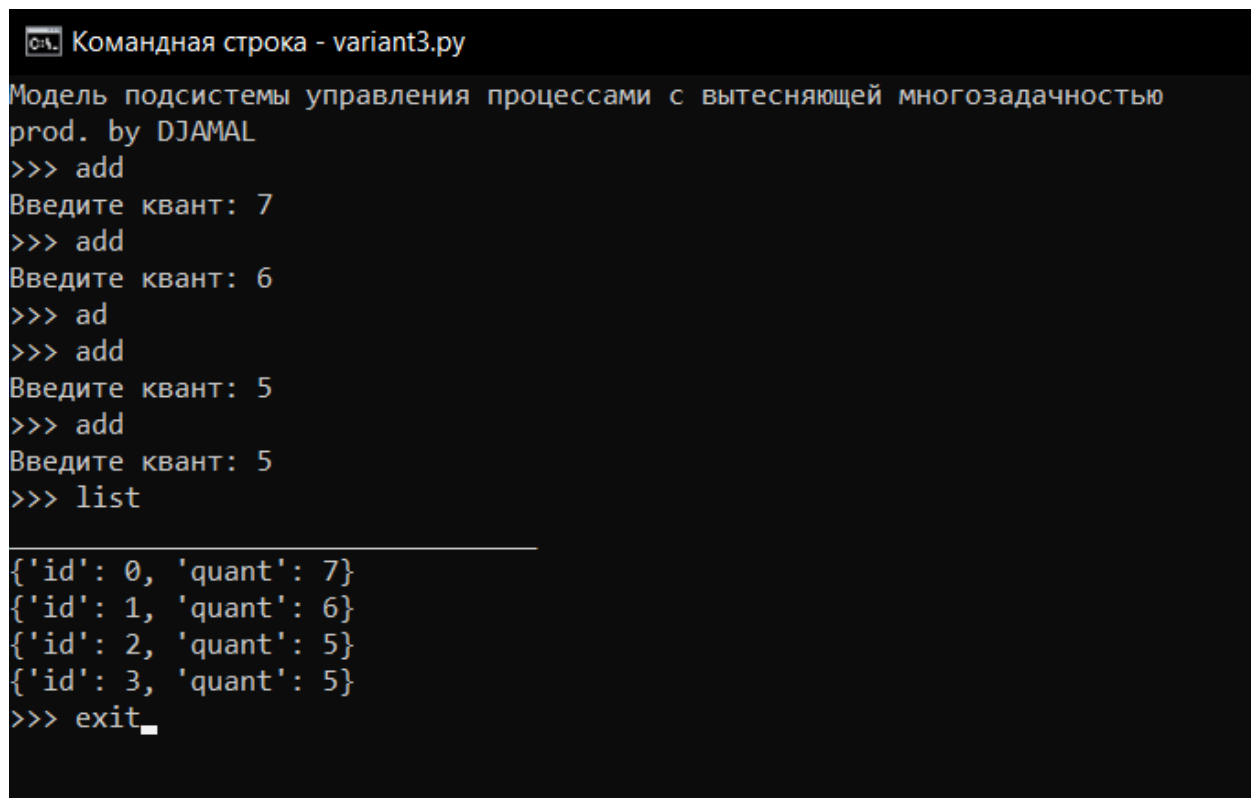


```
E:\Курсовая>variant3.py
Модель подсистемы управления процессами с вытесняющей многозадачностью
prod. by DJAMAL
>>> add
Введите квант: 7
>>> add
Введите квант: 6
>>> ad
>>> add
Введите квант: 5
>>> add
Введите квант: 5
>>> list

{'id': 0, 'quant': 7}
{'id': 1, 'quant': 6}
{'id': 2, 'quant': 5}
{'id': 3, 'quant': 5}
>>> .
```

Рис. 8. Использование команды «list»

Завершить работу программы можно с помощью команды «exit»



```
Командная строка - variant3.py
Модель подсистемы управления процессами с вытесняющей многозадачностью
prod. by DJAMAL
>>> add
Введите квант: 7
>>> add
Введите квант: 6
>>> ad
>>> add
Введите квант: 5
>>> add
Введите квант: 5
>>> list

{'id': 0, 'quant': 7}
{'id': 1, 'quant': 6}
{'id': 2, 'quant': 5}
{'id': 3, 'quant': 5}
>>> exit_
```

Рис. 6. Выход из программы

3.4 Основные характеристики программы

Эта программа, предназначенная для управления процессами пользователем персонального компьютера. Она работает под управлением операционных систем MS Windows 2000/XP/Vista/7/10.

Программа позволяет формировать списки процессов и их приоритетов, и затем выводить результаты их деятельности на экран.

Программа включает в себя набор базовых элементов, каждый из которых позволяет выполнить определенную задачу проекта. После загрузки проекта и запуска процессов значения и параметры работы процессов сразу отображаются на дисплее. Программа работает в штатном режиме, однако в случае необходимости может работать круглосуточно и непрерывно.

Программа прошла всестороннюю проверку и полностью защищена от сбоев и исключений.

4. Программа и методика испытаний

4.1 Объект испытаний

Подсистема управления процессами на основе абсолютных приоритетов.

4.2 Цель испытаний

В процессе испытаний должны быть достигнуты следующие цели и задачи:

При нахождении в потоке нескольких процессов, которые готовы к выполнению, выполняться будут все процессы по мере их поступления.

4.3 Требования к программе

Программа, предназначенная для управления процессами пользователем персонального компьютера. Она работает под управлением операционных систем MS Windows 2000/XP/Vista/7/10.

Позволяет формировать списки процессов и выводить результаты их деятельности на экран с последующим сохранением.

Программа включает в себя набор базовых элементов, каждый из которых позволяет выполнить определенную задачу проекта. После загрузки проекта и запуска процессов значения и параметры работы процессов сразу отображаются на дисплее. Программа работает в штатном режиме, однако в случае необходимости может работать круглосуточно и непрерывно. Программа прошла всестороннюю проверку и полностью защищена от сбоев и исключений

4.4 Методы испытаний

Последовательно в поток программы добавляются процессы с указанием их квантов. После добавления процессов в поток, программа основанная на алгоритме планирования на основе вытесняющей многозадачности выполняет процессы согласно значениям их квантов.

Заключение

Подсистема управления процессами является одной из важнейших частей операционной системы. В этой курсовой работе был представлен один из вариантов его реализации, неприменимый в реальной жизни. Тем не менее, нельзя не отметить, что этот проект весьма упрощен. Подсистема управления процессами, которая реально могла бы стать частью многозадачной операционной системы, требует гораздо большей степени проработанности, уметь работать с прерываниями, разнородными процессами, а к тому же иметь некоторую защиту от внешнего воздействия, ведь умышленное или неумышленное завершение критически важных процессов может привести к краху системы. Но все же, в работе представлена довольно изящная реализация. Главная ее проблема в том, что в ней реализованы не все возможные компоненты, и некоторые моменты регулируются встроенными средствами операционной системой, которая работает на компьютере, в данном случае Windows. При реальном программировании операционной системы, подсистема управления процессами должна быть построена с нуля и иметь определение и описание многих элементов, которые работает в этом проекте по умолчанию

Библиография

1) Безбогов, А.А. Безопасность операционных систем : учебное пособие / А.А. Безбогов, А.В. Яковлев, Ю.Ф. Мартемьянов. – М. : "Издательство Машиностроение-1", 2007. – 220 с.

2) Операционные системы, лекции по операционным системам [Электронный ресурс] / www.osi.ru.ru. - Содержание: Управление процессами; Управление памятью; Управление данными; Управление устройствами

3) Троелсен, Э. С# и платформа .NET: учебное пособие/ Э.Троелсен. – Спб. : "Питер Пресс", 2007. –796с.

4) Мартемьянов, Ю. Ф. Операционные системы. Концепции построения и обеспечения безопасности : учеб. пособие / Ю. Ф. Мартемьянов, Ал. В. Яковлев, Ан. В. Яковлев. – Москва : Горячая линия-Телеком, 2011. – 332 с. : ил., табл. ; 22 см. – (Учебное пособие для высших учебных заведений. Специальность). – Гриф: Доп. УМО. – Библиогр.: с. 325-326.

5) Иртегов, Д. В. Введение в операционные системы : [учеб. пособие] / Дмитрий Иртегов. – СПб. : БХВ-Петербург, 2002. – 624 с. : ил. – Библиогр.: с. 598-607. – Предм.указ.: с. 608-614.

Исходный код программы

```
#!/usr/bin/env python3
# -*- config: utf-8 -*-
```

```
print('Модель подсистемы управления процессами с вытесняющей  
многозадачностью')
```

```
print('prod. by DJAMAL')
```

```
desc_list = []
```

```
dell_list = []
```

```
def CreateProc(quant):
```

```
    description = {  
        'id': len(desc_list),
```

```

        'quant': int(quant)
    }
    desc_list.append(description)

def Start():
    max_quant = 0
    for x in desc_list:
        if x['quant'] > max_quant:
            max_quant = x['quant']
    dell_list = []
    for x in range(1, int(max_quant)+1):
        print('_____ИТЕРАЦИЯ №', x, '_____')
        for y in desc_list:
            y['quant'] -= 1
            print('Выполнение процесса №', y['id'], '. Осталось квантов: ',
y['quant'])
            if y['quant'] == 0:
                print('Процесс №', y['id'], ' завершен')
                dell_list.append(y)

        for q in dell_list:
            desc_list.remove(q)
            dell_list.remove(q)
        print('_____')

while True:
    command = input('>>> ')

```

```
if command == 'add':  
    quant = input('Введите квант: ')  
    CreateProc(quant)  
  
if command == 'start':  
    Start()  
  
if command == 'list':  
    print('_____')  
    for x in desc_list:  
        print(x)  
  
if command == 'exit':  
    break  
  
if command == 'help':  
    print('add - добавление процессов')  
    print('list - вывод списка процессов')  
    print('start - запуск программы')  
    print('help - вывод списка команд')
```