Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Факультет Программной инженерии и компьютерной техники

**Лабораторная работа №1**

по дисциплине «Операционные системы»

**Выполнил**

Ореховский А.,

группа P3317

**Преподаватель**

Лаздин А. В.

Санкт-Петербург

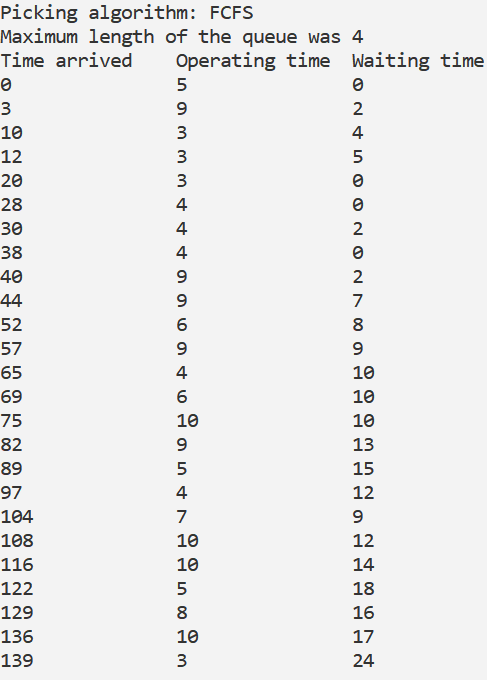
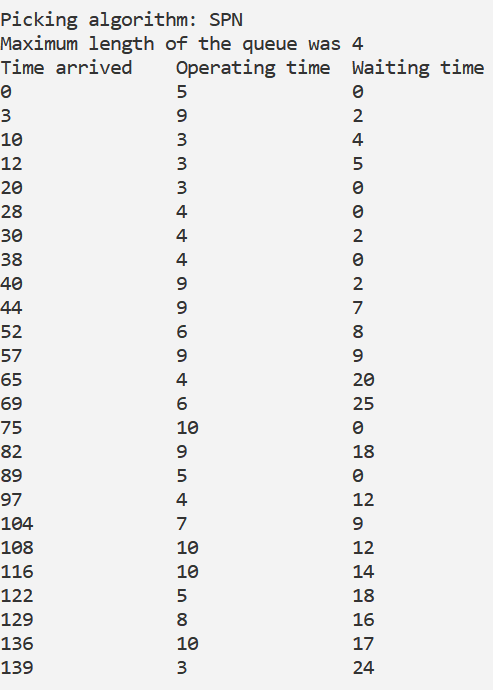
2020

Планирование процессов.  
Задача: изучить основные алгоритмы планирования процессов.

Диапазон длительностей выполнения: [3 .. 10]

Среднее время между поступающими процессами: 5±3

## Набор исходных данных



После вычисления средних времен ожидания для каждого алгоритма, имеются следующие значения:

8,76 – для алгоритма FCFS;

8,26 – для алгоритма SPN.

## Листинг кода

Emulator.cs

public static void Emulate(List<Operation> operations, IAlgorithm picker)

{

// Pseudo proc preparation

var procReleaseAt = 0;

var timePassed = 0;

var maxQueueLength = 0;

// Emulation

while (operations.Where(o => o.TimeLeaved == -1).Count() > 0)

{

if (timePassed >= procReleaseAt)

{

var ind = picker.Pick(operations, timePassed);

if (ind != -1)

{

procReleaseAt = timePassed + operations[ind].OperatingTime;

operations[ind].TimeLeaved = procReleaseAt;

}

}

maxQueueLength = GetMaxQueueLength(operations,

timePassed, maxQueueLength);

timePassed++;

}

// Report purpose

Log(operations, picker, maxQueueLength);

}

FCFSAlgorithm.cs

public int Pick(List<Operation> operations, int timePassed)

{

if (operations.Count == 0)

{

return -1;

}

for (var i = 0; i < operations.Count; i++)

{

if (operations[i].TimeArrived <= timePassed &&

operations[i].TimeLeaved == -1)

{

return i;

}

}

return -1;

}

SPNAlgorithm.cs

public int Pick(List<Operation> operations, int timePassed)

{

var queue = operations.Where( operation =>

operation.TimeArrived <= timePassed &&

operation.TimeLeaved == -1

).ToList();

if (queue.Count == 0)

{

return -1;

}

var minOperatingTime = int.MaxValue;

var ind = 0;

for (var i = 0; i < queue.Count; i++)

{

if (queue[i].TimeArrived >= timePassed &&

minOperatingTime > queue[i].OperatingTime)

{

minOperatingTime = queue[i].OperatingTime;

ind = i;

}

}

return operations.FindIndex(operation =>

operation.TimeArrived == queue[ind].TimeArrived);

}

## Выводы

Сравнив два метода выбора операций, я сделал следующие выводы:

* Несмотря на то, что оба метода в среднем имеют одинаковое время ожидания, алгоритм SPN имеет преимущество над алгоритмом FCFS в определенной ситуации. Если бы мы знали время исполнения следующих процессов, находящихся в состоянии готовность, то могли бы выбрать для исполнения не процесс из начала очереди, а процесс с минимальной длительностью, что в свою очередь уменьшило бы время ожидания в среднем.
* SPN имеет так же недостаток, а именно невозможность знания длительности выполнения операции перед ее выполнением.
* Существует ряд способов определения времени выполнения:
  + Задание вручную программистом
  + Определение, основываясь на предыдущей операции
  + Др.

Каждый из этих способов имеет свои преимущества и недостатки

* Преимущество же FCFS заключается в том, что для его реализации, благодаря его простоте, ничего, кроме очереди, не нужно.