**РУСЕНСКИ УНИВЕРСИТЕТ „АНГЕЛ КЪНЧЕВ“**

**КУРСОВА ЗАДАЧА**

ПО

ОПЕРАЦИОННИ СИСТЕМИ

Студент:

Фак. Номер:

Група:

Специалност:

Дата: Проверил:

**Съдържание**

[1. Задание 3](#_Toc437469361)

[2. Описание на използваните структури данни и алгоритми 3](#_Toc437469362)

[3. Код на програмата (листинг) 4](#_Toc437469363)

[4. Демо 5](#_Toc437469364)

# Задание

**Вариант 5.2.3**

Да се разработи програмна система, симулираща работата на модула за **управление на процесора (стратегия Shortest Job First)** при следните изходни данни:

* ***Основни необходими структури:***
  + Ready таблица на готовите процеси
* ***Основни обработки:***
  + Load стартиране на нов процес;
  + Dispatch извикване на диспечер по стратегия SJF;
  + Finish завършване на изпълняващия се процес
  + Show показване на състоянието на основните структури

Разработеният проект да има подходящ потребителски интерфейс за обръщение към основните обработки. Обяснителната записка да съдържа заданието, описание на използваните структури данни и алгоритми и програмен листинг.

# Описание на използваните структури данни и алгоритми

В началото на програмата се изисква въвеждането на броя на процесите n. След въвеждането му се заделя памет за двата масива (с размер n + 1), съответно масив за съхраняване на времето на пристигане (at – arrival time) и масив за съхраняване на времето, нужно за изпълняването на всеки един от процесите (bt – burst time). Данните за масивите се въвеждат от конзолата, a в една променлива се сумира времето за изпълнение.

След което:

1. първо се намира процесът, който е пристигнал най-рано и е с най-малко време за изпълнение и се изпълнява докрай;
2. след като процесът завърши изпълнението си се избира един от процесите, които са се появили по време на изпълнението му, и е с най-малко време за изпълнение (burst time) и се изпълнява докрай. В противен случай се изпълнява стъпка 1).
3. Повтаря се стъпка 2), докато всички процеси бъдат завършени.

Времето на изчакване (Tw) на процесите се пресмята по следния начин: от времето на стартиране на процеса се изважда времето му на пристигане.

Оборотното време (Tr) на процесите се пресмята като от към времето на изчакване (Tw) се добави времето на изпълнение (bt).

Накрая се пресмята и средното време на изчакване и средното оборотно време: сумата съответно на времето на изчакване и оборотното време на процесите се разделя на броя на процесите.

Преди края на програмата се освобождава и динамичната памет, заделена за масивите.

# Код на програмата (листинг)

#include <stdio.h>

#include <malloc.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "bgr");

int time,

\*bt,

\*at,

sum\_bt = 0,

smallest,

n,

i;

int sum\_turnaround = 0,

sum\_wait = 0;

printf("Въведете броя на процесите : ");

scanf("%d", &n);

at = (int\*)malloc(sizeof(int)\*n+1);

bt = (int\*)malloc(sizeof(int)\*n+1);

for (i = 0; i < n; i++)

{

printf("\nВреме на пристигане на P%d : ", i + 1);

scanf("%d", &at[i]);

printf("Време за изпълнение на P%d : ", i + 1);

scanf("%d", &bt[i]);

sum\_bt += bt[i];

}

bt[n + 1] = 9999;

printf("\n\nПроцес\t|\tTw\t|\tTr\n\n");

for (time = 0; time < sum\_bt;)

{

smallest = n + 1;

for (i = 0; i < n; i++)

{

if (at[i] <= time && bt[i] > 0 && bt[i] < bt[smallest])

smallest = i;

}

if (smallest == n + 1)

{

time++;

continue;

}

printf("P[%d]\t|\t%d\t|\t%d\n", smallest + 1, time - at[smallest], time + bt[smallest] - at[smallest]);

sum\_turnaround += time + bt[smallest] - at[smallest];

sum\_wait += time - at[smallest];

time += bt[smallest];

bt[smallest] = 0;

}

printf("\n\nСредно време на изчакване = %.2f", (float)sum\_wait / n);

printf("\nСредно оборотно време = %.2f\n", (float)sum\_turnaround / n);

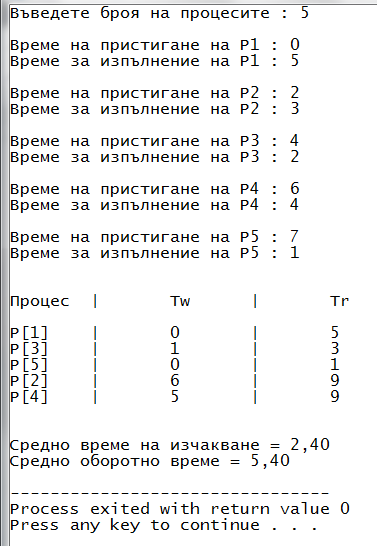
free(at);

free(bt);

return 0;

}

# Демо



Фигура 1 Тестов пример 1