**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

по дисциплине «Операционные системы»

Программирование планировщиков процессов

Выполнил студент

группы ИТИ-11

*Громыко И. В.*

Проверил преподаватель

*Карась О.В.*

Гомель 2022

**Цель работы:** разработать программу, осуществляющую моделирование режима работы с разделение времени.

**Задание.**

В соответствии с вариантом (выдается преподавателем), разработать программу на языке С# (или другом), осуществляющую моделирование работы планировщика процессов. Моделирование режима разделения времени выполняется в соответствии с заданием (таблица 1).

На экран выводится следующая информация: 1. Номер текущего кванта времени процессора; 2. Таблица процессов с указанием имени процессов, продолжительности, приоритета (в зависимости от задания), оставшегося времени выполнения, время появления; 3. Таблица планирования процессов с отображением текущего состояния процессов. После запуска, программа должна диалоговом режиме, ввести информацию процессах – имя, длительность, приоритет, время появления. Выполнение должно производиться в пошаговом режиме (по нажатию на кнопку).

|  |  |
| --- | --- |
| Вариант | Алгоритм планировщика |
| 6 | Алгоритм SJF. Не вытесняющий, приоритетный |

**Ход работы**

**Порядок выполнения задания 1.**

На рисунке 1 показано выполнение задания.

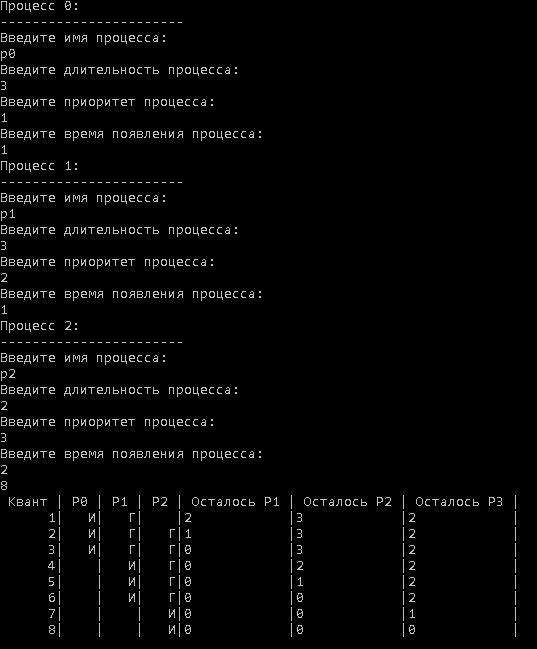


Рисунок 1 – Выполнение задания

**Вывод:** в процессе работы разработали планировщик процессов определённого алгоритма.

**ПРИЛОЖЕНИЕ А**

**Листинг программы**

**Листинг задания:**

#include <stdio.h>

#include <conio.h>

#include <string.h>

#include <locale.h>

#include <stdlib.h>

char name1[10],name2[10],name3[10];

int duration1, duration2, duration3;

int prior1,prior2,prior3;

int time1, time2, time3;

struct processes

{

char name[10];

int duration;

int prior;

int time;

char symb[5];

int left;

};

int main()

{

int minPriority,minDuration,currentInd;

setlocale(LC\_ALL,"RUS");

struct processes process[3];

for (int i=0;i<3;i++)

{

printf("Процесс %d: \n-----------------------\n",i);

fflush(stdin);

puts("Введите имя процесса: ");

gets(process[i].name);

puts("Введите длительность процесса: ");

scanf("%d",&process[i].duration);

puts("Введите приоритет процесса: ");

scanf("%d",&process[i].prior);

puts("Введите время появления процесса: ");

scanf("%d",&process[i].time);

process[i].left=process[i].duration;

}

int timeAll=process[0].duration+process[1].duration+process[2].duration;

printf("%d\n",timeAll);

printf(" Квант |%4s|%4s|%4s| Осталось P1 | Осталось P2 | Осталось P3 |\n",process[0].name,process[1].name,process[2].name);

for (int i=1;i<=timeAll;i++)

{

minPriority=100;

minDuration=timeAll;

for(int k=0;k<3;k++)

{

if (process[k].time<=i && process[k].left>0)

{

if (process[k].prior<=minPriority)

{

minPriority=process[k].prior;

if (process[k].duration<minDuration)

{

minDuration=process[k].duration;

currentInd=k;

}

}

}

}

for (int k=0;k<3;k++)

{

if(currentInd==k && process[k].left>0)

strcpy(process[k].symb,"И");

else if(process[k].time<=i && process[k].left>0)

strcpy(process[k].symb,"Г");

else

strcpy(process[k].symb," ");

}

//printf("%7d|%4s|%4s|%4s|\n",i,process[0].symb,process[1].symb,process[2].symb);

fflush(stdin);

process[currentInd].left--;

printf("%7d|%4s|%4s|%4s|%-13d|%-13d|%-13d|",i,process[0].symb,process[1].symb,process[2].symb,process[0].left,process[1].left,process[2].left);

// printf("\n");

getchar();

}

fflush(stdin);

getchar();

return 0;

}