**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 № 11

по дисциплине «Операционные системы»

Управление файловой системой

Выполнил студент

группы ИТИ-11

*Громыко И. В.*

Проверил преподаватель

*Карась О.В.*

Гомель 2022

**Цель работы:** разработать программу с использованием дополнительных утилит.

**Задание.**

Разработать программу с использованием вспомогательной утилиты.

Таблица 1

|  |  |
| --- | --- |
| Вариант | Дополнительная утилита |
| 6 | Кэширование |

**Ход работы**

**Порядок выполнения задания.**

На рисунках 1-4 показаны результаты выполнения программы.

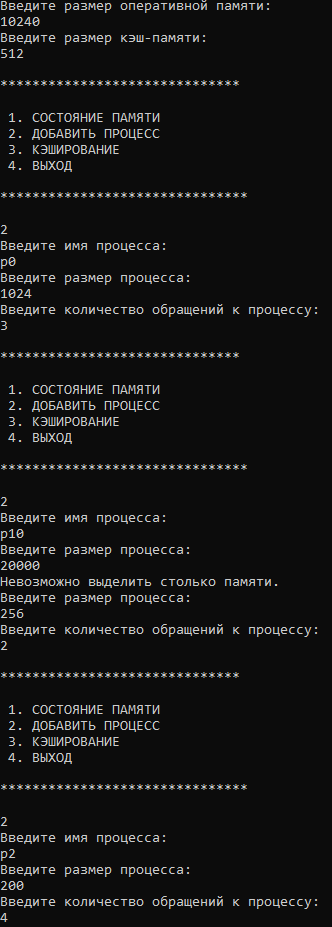


Рисунок 1 – Результат выполнения программы

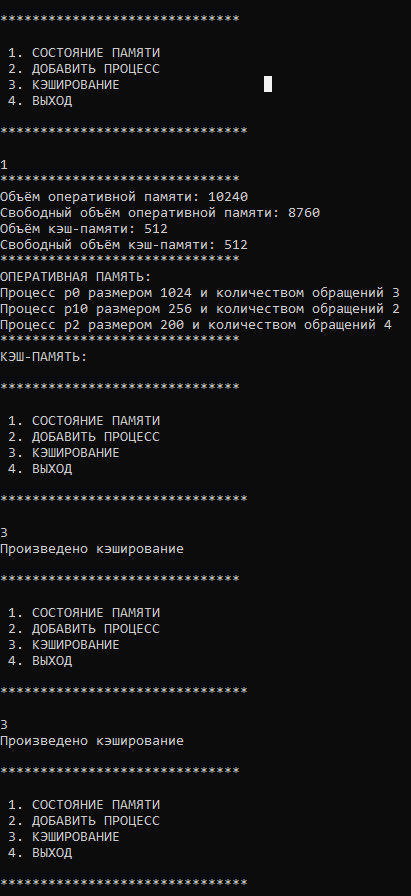


Рисунок 2 – Результат выполнения программы

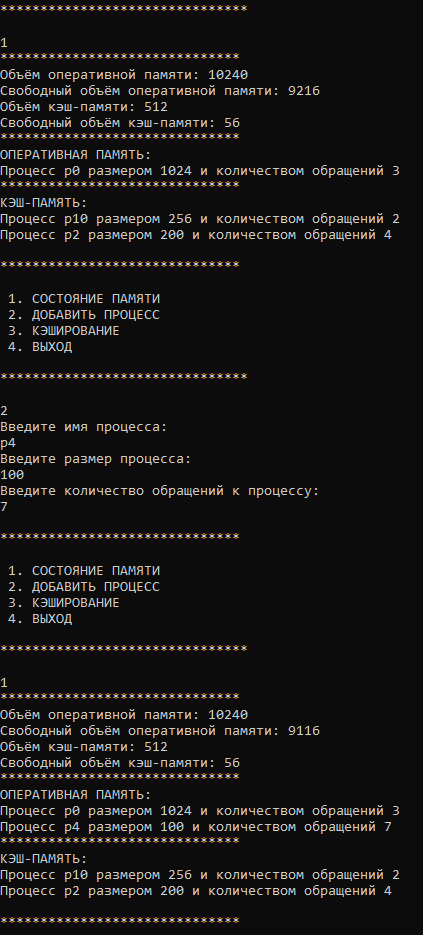


Рисунок 3 – Результат выполнения программы

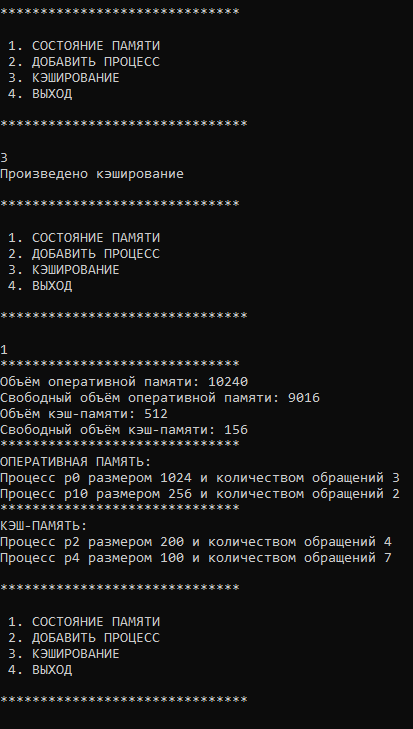


Рисунок 4 – Результат выполнения программы

**Вывод:** в процессе работами ознакомились с разлиными дополнительными утилитами, разработали программу с её применением.

**ПРИЛОЖЕНИЕ А**

**Листинг программы**

**Листинг программы laba:**

#include <stdio.h>

#include <string.h>

#include <locale.h>

#include <stdlib.h>

int cprocesses=0;

int fullMemory=0,freeMemory=0;

int cashMemory=0,freeCashMemory=0;

struct processes

{

char name[30];

int cash;

int size;

int request;

};

processes process[50];

void showData()

{

puts("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("Объём оперативной памяти: %d\n",fullMemory);

printf("Свободный объём оперативной памяти: %d\n",freeMemory);

printf("Объём кэш-памяти: %d\n",cashMemory);

printf("Свободный объём кэш-памяти: %d\n",freeCashMemory);

puts("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

puts("ОПЕРАТИВНАЯ ПАМЯТЬ: ");

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

{

if(process[i].cash==2)

{

printf("Процесс %s размером %d и количеством обращений %d\n",process[i].name,process[i].size,process[i].request);

}

}

puts("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

puts("КЭШ-ПАМЯТЬ: ");

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

{

if(process[i].cash==1)

{

printf("Процесс %s размером %d и количеством обращений %d\n",process[i].name,process[i].size,process[i].request);

}

}

}

void cash()

{

int t=1,maxRequest=0,maxRequestIndex,count=0;

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

{

if (process[i].cash==2 && process[i].request>maxRequest && process[i].size<freeCashMemory)

{

maxRequest=process[i].request;

maxRequestIndex=i;

count++;

}

}

if (count)

{

process[maxRequestIndex].cash=1;

freeMemory+=process[maxRequestIndex].size;

freeCashMemory-=process[maxRequestIndex].size;

}

else

{

count=0;

int minCashRequest=0,minCashRequestIndex=0;

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

{

if (process[i].cash==1)

{

minCashRequest=process[i].request;

minCashRequestIndex=i;

break;

}

}

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

{

if (process[i].cash==1 && process[i].request<=minCashRequest && process[i].size<=freeMemory)

{

minCashRequest=process[i].request;

minCashRequestIndex=i;

}

}

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

{

if (process[i].cash==2 && process[i].request>minCashRequest && process[i].size<=process[minCashRequestIndex].size && process[i].size<cashMemory)

{

maxRequest=process[i].request;

maxRequestIndex=i;

count++;

}

}

if (count)

{

process[minCashRequestIndex].cash=2;

process[maxRequestIndex].cash=1;

freeCashMemory-=process[maxRequestIndex].size;

freeCashMemory+=process[minCashRequest].size;

freeMemory+=process[maxRequestIndex].size;

freeMemory-=process[minCashRequest].size;

t=1;

}

else

t=0;

}

if(t)

puts("Произведено кэширование");

else

puts("Невозможно произвести кэширование");

}

void addProcess()

{

int size,t=1;

int min2Cash=cashMemory,tmp; // fullMemory=3072 freeMemory=2048 cashMemory=612 freeCashMemory=56

char name[30];

fflush(stdin);

puts("Введите имя процесса: ");

gets(name);

while (t)

{

fflush(stdin);

puts("Введите размер процесса: ");

scanf("%d",&size);

if(size>freeMemory)

{

puts("Невозможно выделить столько памяти.");

}

else

{

strcpy(process[cprocesses].name,name);

process[cprocesses].size=size;

freeMemory-=size;

puts("Введите количество обращений к процессу: ");

scanf("%d",&process[cprocesses].request);

t=0;

process[cprocesses].cash=2;

}

}

cprocesses++;

}

int main()

{

setlocale(LC\_ALL,"RUS");

puts("Введите размер оперативной памяти: ");

scanf("%d",&fullMemory);

freeMemory=fullMemory;

puts("Введите размер кэш-памяти: ");

scanf("%d",&cashMemory);

freeCashMemory=cashMemory;

int t=1,menu;

while (t)

{

puts("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

puts(" 1. СОСТОЯНИЕ ПАМЯТИ");

puts(" 2. ДОБАВИТЬ ПРОЦЕСС");

puts(" 3. КЭШИРОВАНИЕ");

puts(" 4. ВЫХОД\n");

puts("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

scanf("%d",&menu);

switch(menu)

{

case 1:

showData();

break;

case 2:

addProcess();

break;

case 3:

cash();

break;

case 4:

t=0;

break;

default:

printf("Неверно введённые данные\n");

}

}

fflush(stdin);

getchar();

return 0;

}