

**Министерство образования и науки Российской Федерации**

**Федеральное агентство по образованию**

**Государственное образовательное учреждение высшего профессионального образования**

**«Московский государственный технический университет имени Н.Э. Баумана»**

**(МГТУ им. Н.Э. Баумана)**

**Факультет «Робототехника и комплексная автоматизация» (РК)**

**Кафедра «Системы автоматизированного проектирования» (РК6)**

****

**Отчет по лабораторной работе №4 по курсу**

**«Операционные системы»**

**Студент: Сергеева Диана**

**Группа:** РК6-56Б

**Преподаватель:** Грошев С.В.

Проверил:

Дата:

2021 год

**13 вариант**

**Задание «Конкуренция параллельных процессов OS UNIX»:**

Требуется разработать программу, которая визуально демонстрирует поведение параллельных процессов OS UNIX на основе механизма диспетчеризации. С этой целью программа должна порождать заданное число процессов, которые должны выполняться параллельно, конкурирую за ресурсы процессора. При этом каждому порожденному процессу должна быть сопоставлена символьная метка в виде заглавной латинской буквы и номер строки консоли. Происходит посимвольное стирание заданного числа одинаковых строк параллельными процессами слева-направо. Число и состав строк передаются аргументами командной строки. При разработке программы необходимо применить системные вызовы fork, exit, wait, usleep, write, также использовать Escape-последовательности управления курсором.

**Исходный код:**

#include <unistd.h>

#include <stdio.h>

#include <sys/timeb.h>

/\* clear standart text mode window \*/

void clrscr(char \* str, int num)

{

unsigned char esc[11];

esc[0] = 27;

esc[1] = '[';

/\* obtain set cursor to left-top window corner \*/

esc[2] = 'H';

write(1, esc, 3);

/\* obtain clear escape sequence \*/

esc[2] = '2';

esc[3] = 'J';

write(1, esc, 4);

//----------------

int length = sprintf(str,"%s", str);

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

{

write(1, str, length);

printf("\n\r");

//esc[2] = 'B';

//write(1, esc, 3);

}

//----------------

/\* obtain set cursor to left-top window corner \*/

esc[2] = 'A';

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

write(1, esc, 3);

return;

}

/\* position cursor in standart text window \*/

void gotoxy(int tx, int ty, char c)

{

unsigned char esc[16];

static unsigned char ystr[3]; /\* vertical cursor location \*/

static unsigned char xstr[3]; /\* horizontal cursor location \*/

int i; /\* ESC-sequence current index \*/

int j; /\* cursor location current index \*/

/\* convert cursor location to text format \*/

if ((tx > 99) || (ty > 99))

tx = ty = 99;

if ((tx < 1) || (ty < 1))

tx = ty = 1;

xstr[0] = xstr[1] = xstr[2] = '\0';

ystr[0] = ystr[1] = ystr[2] = '\0';

sprintf((char \*)xstr, "%d", tx);

sprintf((char \*)ystr, "%d", ty);

/\* obtain goto escape sequence \*/

esc[0] = 27;

esc[1] = '[';

i = 2;

j = 0;

while (ystr[j])

esc[i++] = ystr[j++];

j = 0;

esc[i++] = ';';

while (xstr[j])

esc[i++] = xstr[j++];

esc[i++] = 'H';

esc[i++] = '\b';

esc[i++] = ' ';

esc[i++] = c;

esc[i] = '\0';

write(1, esc, i);

//printf("%c", c);

return;

}

int main(int argc, char \*argv[])

{

int PROCNUM = atoi(argv[1]);

int dist = atoi(argv[2]);

char\* str = argv[3];

int p;

int x = 1;

int status;

int i;

int j = 0;

int \*pid = sbrk((PROCNUM + 1) \* sizeof(int));

char \*lead = sbrk((PROCNUM + 1) \* sizeof(char));

char bell = '\007';

struct timeb tp[1];

int jump;

clrscr(argv[3], PROCNUM);

dist++;

while (j < PROCNUM)

{

if ((pid[j] = fork()) == 0)

{

usleep(PROCNUM - j);

while (x < dist)

{

gotoxy(x, j + 1, ' ');

ftime(tp);

if ((tp[0].millitm % (j+'A')) != j)

continue;

x++;

for (i = 0; i < 100000000; i++)

;

}

exit('A' + j);

}

j++;

}

j = 0;

while ((p = wait(&status)) != (-1))

{

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

if (pid[i] == p)

lead[j++] = (char)((status >> 8) & '\377');

write(1, &bell, 1);

}

lead[j] = '\n';

sleep(1);

gotoxy(1, PROCNUM + 3, '\n');

write(1, lead, PROCNUM + 1);

exit(0);

}

**Результат работы программы:**





