

1	2	3	4
+	±	✓	✓

Контрольная работа №3.

Вариант 1. (ФИО: Казыриг М.И.)

Задание 1. Написать программу.

Организовать у процесса-«отца» двух «сыновей». Старший сын должен уметь выводить на экран предложение «*Воображение важнее.*». Младший - «*чем знание.*». В совместной работе сыновья должны вывести в stdout 10 раз сообщение: «*Воображение важнее, чем знание.*», - и умереть. Отец должен «умереть» после «смерти» сыновей.

Замечания: Механизм организации «смерти» произвольный. Синхронизацию работы «сыновей» реализовать с помощью **каналов**.

Задание 2. Написать программу.

После запуска программы процесс должен начать бесконечно выводить строку: «*Значимость программы*». В случае нажатия пользователем на клавиатуре «Ctrl+C» программа должна вывести строку «*обратно пропорциональна*». И начать бесконечно выводить строку «*объему выводимых ею результатов*». В случае повторного нажатия на «Ctrl+C» процесс должен умереть от полученного **сигнала**.

Задание 3. Написать две программы.

Программа-чтец должна организовать **IPC очередь сообщений**, передать в неё содержимое текстового файла «PeaceAndWar.txt» построчно и завершить своё исполнение.

Программа-писарь должна прочитать сообщения из общей с «чтецом» очереди сообщений, вывести их на экран и закончить своё исполнение.

Примечание: работу с файлом реализовать любым удобным Вам образом.

Задание 4. Написать две программы.

Первая программа должна выводить на экран заглавные буквы английского алфавита от 'Z' до 'A' в обратном алфавитном порядке.. Вторая программа – прописные от 'z' до 'a' (в обратном алфавитном порядке). С помощью **семафоров IPC** организовать синхронизацию выполнения программ так, чтобы на экран заглавные и прописные буквы выводились строго по очереди.

Например, правильно: Zz Yy Xx ... Bb Aa

Неправильно: ZY zy xX ... Bb aA

ФИО: Жазыриг Максим

№2

```
int old=1, new=0;
void Фрз(int sigNum)
{ signal(SIGINT, Фрз);
  new++;
}
```

```
int main()
{
```

```
  signal(SIGINT, Фрз);
```

```
  while (old != new)
```

```
  { printf("Значимость программы");
```

```
  }
```

```
  while(1){
```

```
    printf("одному выводится его результатов");
```

```
  }
```

```
  return 0;
```

3

№1

```
int main()
{
```

```
  int fd1[2], fd2[2], i=0, k, j;
```

```
  pipe(fd1);
```

```
  pipe(fd2);
```

```
  if (fork() == 0)
```

```
  { close(fd1[0]); close(fd2[1]);
```

```
    while (i < 10)
```

```
    { printf("Возвращение вамее");
```

```
      i++;
```

```
      write(fd1[1], &i, sizeof(int));
```

```
      read(fd2[0], &k, sizeof(int));
```

```
    }
```

```
  close(fd1[1]);
```

```
  close(fd2[0]);
```

```
  exit(0);
```

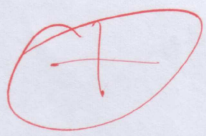
```
  } else
```



```

if (fork() == 0)
{
    close(fd2[0]);
    close(fd1[1]);
    while (i < 10)
    {
        read(fd1[0], &j, sizeof(int));
        printf("rem zmanue.\n");
        write(fd2[1], &i, sizeof(int));
    }
    close(fd2[1]);
    close(fd1[0]);
    exit(10);
} else // not eg
{
    close(fd1[0]);
    close(fd1[1]);
    close(fd2[0]);
    close(fd2[1]);
    wait();
    wait();
}
return 0;
}

```



He bozgenere uallitb,
NO3

segmentation fault

```

// key
int main()
{
    int id1, id2;
    char *string;
    struct BufE *PS;
    PS = (struct BufE *) malloc(256 * sizeof(char));
    id1 = fork("user/bmla.txt", 'a');
    msgget(id1, 0);
    id2 = open("peaceandwar.txt", O_RDONLY, 0);
    while (fgets(PS.string, 256, id2) > 0)
    {
        msgsnd(id1, PS, 1, 0);
    }
    close(id2);
    msync(id1, 0, MS_INVALID);
    return 0;
}

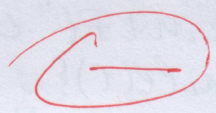
```

CHONBUO!

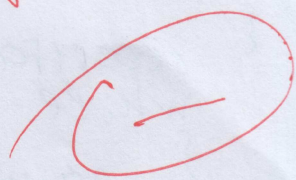
```

// mncapb
int main()
{
    int id1;
    struct BufE *PS;
    PS.string = (char *) malloc(256 * sizeof(char));
    id1 = fork("user/bmla.txt", 'a');
    msgget
    while (msgrecv(id1, PS, 0) > 0)
    {
        printf("%s", PS.string);
    }
    return 0;
}

```



He magrotobunulo
 k IPC



NO4

```

int main()
{
    int id1, s = 1;
    char c = 'Z';
    id1 = fork("user/bmla.txt", 'a');
    msgget(id1, 1, 0);
    while (c != 'A')
    {
        semop(1, s, 1);
        printf("%c", c);
        c = c - 1;
        semop(1, s, -1);
    }
}

```

```

int main()
{
    int id1;
    char c = 'Z';
    id1 = fork("user/bmla.txt", 'a');
    while (c != 'A')
    {
        semop(1, s, 1);
        printf("%c", c);
        c = c - 1;
        semop(1, s, -1);
    }
}

```

semopet?

ФИО: Черновин

№2

```

int main()
{
    int old=1, new=0;
    while (old != new)
    signal(SIGINT, FPE);
    while (old != new)
    {
        printf("Значимость программы ");
        printf("обратно пропорциональна");
        signal(SIGINT, SIG_IGN);
        while(1)
        {
            printf("Обсчету вводимых результатов");
        }
    }
    return 0;
}
    
```

```

void fpe()
{
    new++;
}
    
```

№1

```

int main()
{
    int fd[2], fd2[2];
    int k=0, id1, id2;
    char c;
    pipe(fd); c='a';
    pipe(fd2);
    if ((id1 = fork()) == 0)
    {
        close(fd[0]); close(fd2[1]);
        printf("Воспроизведе символ");
        write(fd[1], c, 1);
        read(fd2[0], c, 1);
        close(fd[1]); close(fd2[0]);
        exit(0);
    }
    else
    {
        if ((id2 = fork()) == 0)
        {
            close(fd2[0]); close(fd1[1]);
        }
    }
}
    
```

```

while (i < 10)
while (read(fd2[0], c) !=
else
{
    wait();
    wait();
}
return 0;
    
```


Way 3

```

int main()
{
  int id1;
  char *string;
  string = (char *) malloc(256 * sizeof(char));
  id1 = fork("a.txt", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
  msgget(id1, 0666 | 0_CREAT);
  fd = open("peace and war.txt", O_RDONLY);
  while (fgets(string, 256, fd))
    msgsnd(id1, string, 1);
  close(id1);
  return 0;
}

```

```

int main()
{
  int id1;
  char *string;
  string = (char *) malloc(256 * sizeof(char));
  id1 = fork("a.txt", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
  msgget(id1, 0666);
  while (msgrecv(string, id1, string, 1))
    printf("%s", string);
  return 0;
}

```

Way 4

```

int main()
{
  int id1;
  char c = 'z';
  id1 = fork("a.txt", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
  while (c != 'A')
  {
    send(id1, &c, 1, 0);
    printf("%c", c);
    c = c - 1;
  }
  return 0;
}

```

```

int main()
{
  int id1;
  char c = 'z';
  id1 = fork("a.txt", 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0);
  while (c != 'a')
  {
    recv(id1, &c, 1, 0);
    printf("%c", c);
    c = c - 1;
  }
  return 0;
}

```