Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №4
по дисциплине: **ОСИСП**
Тема: GCC. ПРОЦЕССЫ

Выполнил
студент 2 курса
Корнасевич И. Д.

Проверил
Давидюк Ю. И.

Брест 2021

Задание для выполнения Написать программу, которая будет реализовывать следующие функции:

- сразу после запуска получает и сообщает свой ID и ID родительского процесса
- перед каждым выводом сообщения об ID процесса и родительского процесса эта информация получается заново
- порождает процессы, формируя генеалогическое дерево согласно варианту, сообщая, что процесс с ID таким-то породил процесс с таким-то ID
- перед завершением процесса сообщить, что процесс с таким-то ID и таким-то ID родителя завершает работу
- один из процессов должен вместо себя запустить программу, указанную в варианте задания

На основании выходной информации программы предыдущего пункта изобразить генеалогическое дерево процессов (с указанием идентификаторов процессов), объяснить каждое выведенное сообщение и их порядок в предыдущем пункте.

main.c

```c
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

void created(int procNum){
    printf("Created %d: parent = %d, this = %d\n", procNum, getppid()
   , getpid());
}

void exited(int procNum){
    printf("Exited %d: parent = %d, this = %d\n", procNum, getppid(),
    getpid());
}


int main() {

    created(0);

    pid_t pid;
    int procNum = 1;
    if ((pid = fork()) == 0){
        procNum = 1;
        created(procNum);

        if ((pid = fork()) == 0){
            procNum = 4;
            created(procNum);
            exited(procNum);
            exit(0);
        } else {
```

```
30              sleep(1);
31          }
32
33          if ((pid = fork()) == 0){
34              procNum = 5;
35              created(procNum);
36              exited(procNum);
37              exit(0);
38          } else {
39              sleep(1);
40          }
41
42          exited(procNum);
43          exit(0);
44      } else {
45          sleep(3);
46      }
47
48      if ((pid = fork()) == 0){
49          procNum = 3;
50          created(procNum);
51
52          if ((pid = fork()) == 0){
53              procNum = 6;
54              created(procNum);
55
56              if ((pid = fork()) == 0){
57                  procNum = 7;
58                  created(procNum);
59                  execl("/bin/pwd","pwd",NULL);
60                  exited(procNum);
61                  exit(0);
62              } else {
63                  sleep(1);
64              }
65
66              exited(procNum);
67              exit(0);
68          } else {
69              sleep(2);
70          }
71
72          exited(procNum);
73          exit(0);
74      } else {
75          sleep(3);
76      }
77
78      exited(0);
79      exit(0);
80  }
```

results.txt

```
1  Created 0: parent = 4316, this = 6074
2  Created 1: parent = 6074, this = 6075
```

```
3  Created 4: parent = 6075, this = 6076
4  Exited 4: parent = 6075, this = 6076
5  Created 5: parent = 6075, this = 6081
6  Exited 5: parent = 6075, this = 6081
7  Exited 1: parent = 6074, this = 6075
8  Created 3: parent = 6074, this = 6082
9  Created 6: parent = 6082, this = 6083
10 Created 7: parent = 6083, this = 6084
11 /home/ivan/Labs/latex/OSISP/lab4/src/cmake-build-debug
12 Exited 6: parent = 6082, this = 6083
13 Exited 3: parent = 6074, this = 6082
14 Exited 0: parent = 4316, this = 6074
```

Дерево процессов:

- 0
  - 1
    - 4
    - 5
  - 3
    - 6
      - 7

Вывод: Я познакомился со способами управления жизненным циклом процессов при помощи функций fork(), execl(), exit(). Также узнал, что такое pid и ppid.