**Задание 5:**

В программу с программным каналом включить собственный обработчик сигнала. Использовать сигнал для изменения хода выполнения программы.

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <unistd.h>
4. #include <string.h>
5. #include <signal.h>
6. #include <time.h>
7. #include <sys/wait.h>
9. #define LEN 100
10. #define DELAY 2
12. **int** p\_flag = 0;
14. **void** catch\_sig(**int** signum) {
15. printf("Proccess %d: catched signal %d\n", getpid(), signum);
16. p\_flag = 1;
17. }
19. **int** main() {
20. //Pipe
21. **int** fd[2];
22. **if** (pipe(fd) == -1)
23. {
24. perror("Couldn't pipe.");
25. exit(1);
26. }
27. // Ctrl + Z
28. **void** (\*old\_handler)(**int**) = signal(SIGTSTP, catch\_sig);
30. // Child 1
31. pid\_t child1 = fork();
32. **if** (child1 == -1)
33. {
34. perror("Couldn't fork.");
35. exit(1);
36. }
37. **else** **if** (child1 == 0)
38. {
39. **while** (!p\_flag) ;
40. close(fd[0]);
41. **char** msg1[] = "Hello From Child 1 ";
42. write(fd[1], msg1, LEN);
43. printf("Child 1: writen [%s] to Pipe\n", msg1);
44. }
46. // Child 2
47. pid\_t child2 = fork();
48. **if** (child2 == -1)
49. {
50. perror("Couldn't fork.");
51. exit(1);
52. }
53. **else** **if** (child2 == 0)
54. {
55. **while** (!p\_flag) ;
56. close(fd[0]);
57. **char** msg2[] = "Hello From Child 2 ";
58. write(fd[1], msg2, LEN);
59. printf("Child 2: writen [%s] to Pipe\n", msg2);
60. }
61. // Parent
62. **if** (child1 != 0 && child2 != 0)
63. {
64. printf("Parent: pid = %d\n", getpid());
65. printf("Child 1: pid = %d\n", child1);
66. printf("Child 2: pid = %d\n\n", child2);
67. printf("Parent: waiting for CTRL+Z signal\n");
68. **while** (!p\_flag);
70. **int** status1, status2;
71. pid\_t ret1 = wait(&status1);
72. pid\_t ret2 = wait(&status2);
74. printf("------------\n");
75. close(fd[1]);
76. **char** msg1[LEN], msg2[LEN];
77. read(fd[0], msg1, LEN);
78. read(fd[0], msg2, LEN);
79. printf("Parent: read from Pipe [%s%s]\n", msg1, msg2);
80. **if** (WIFEXITED(status1))
81. printf("Parent: child %d finished with %d code.\n", ret1, WEXITSTATUS(status1));
82. **else** **if** (WIFSIGNALED(status1))
83. printf("Parent: child %d finished from signal with %d code.\n", ret1, WTERMSIG(status1));
84. **else** **if** (WIFSTOPPED(status1))
85. printf("Parent: child %d finished from signal with %d code.\n", ret1, WSTOPSIG(status1));
87. **if** (WIFEXITED(status2))
88. printf("Parent: child %d finished with %d code.\n", ret2, WEXITSTATUS(status2));
89. **else** **if** (WIFSIGNALED(status2))
90. printf("Parent: child %d finished from signal with %d code.\n", ret2, WTERMSIG(status2));
91. **else** **if** (WIFSTOPPED(status2))
92. printf("Parent: child %d finished from signal with %d code.\n", ret2, WSTOPSIG(status2));
93. }
94. signal(SIGTSTP, old\_handler);
95. **return** 0;
96. }

