Kamil Borkowski WCY22IY1S1 83374

Zad 1 FORK

Kod źródłowy:

#include <stdio.h>

#include <unistd.h>

#include <stdlib.h>

int main()

{//jesli w case 0 jest pid\_t to tworzymy potomka, jesli w default jest pid\_t to tworzymy wezel tego samego poziomu co w case 0

//wlaczenie programu wywoluje P1

pid\_t pid = fork();

switch (pid)

{

case 0:

/\* Potomny proces P2 \*/

printf("P2: id: %d id\_przodka: %d\n", getpid(), getppid());//pierwszy potomek P1

exit(0);

case -1:

printf("Blad P1\n");

exit(1);

default:

/\* Proces macierzysty P1 \*/

printf("P1: id: %d id\_przodka: %d\n", getpid(), getppid());

pid\_t pid2 = fork();

switch (pid2)

{

case 0:

/\* Potomny proces P3 \*/

printf("P3: id: %d id\_przodka: %d\n", getpid(), getppid());//drugi potomek P1

pid\_t pid6 = fork();

switch (pid6)

{

case 0:

/\* Potomny proces P6 \*/

printf("P6: id: %d id\_przodka: %d\n", getpid(), getppid());//pierwszy potomek P3

exit(0);

case -1:

printf("Blad P6\n");

exit(1);

default:

wait(NULL);//wait dla P6

pid\_t pid7 = fork();

switch (pid7)

{

case 0:

/\* Potomny proces P7 \*/

printf("P7: id: %d id\_przodka: %d\n", getpid(), getppid());//drugi potomek P3

exit(0);

case -1:

printf("Blad P7\n");

exit(1);

default:

wait(NULL);//wait dla P7

pid\_t pid8 = fork();

switch (pid8)

{

case 0:

/\* Potomny proces P8 \*/

printf("P8: id: %d id\_przodka: %d\n", getpid(), getppid());//trzeci potomek P3

pid\_t pid9 = fork();

switch (pid9)

{

case 0:

/\* Potomny proces P9 \*/

printf("P9: id: %d id\_przodka: %d\n", getpid(), getppid());//pierwszy potomek P8

exit(0);

case -1:

printf("Blad P9\n");

exit(1);

default:

wait(NULL);//wait dla P9

pid\_t pid10 = fork();

switch (pid10)

{

case 0:

/\* Potomny proces P10 \*/

printf("P10: id: %d id\_przodka: %d\n", getpid(), getppid());//drugi potomek P8

exit(0);

case -1:

printf("Blad P10\n");

exit(1);

default:

wait(NULL);//wait dla P10

exit(0);

}

}

case -1:

printf("Blad P8\n");

exit(1);

default:

wait(NULL);//wait dla P8

exit(0);

}

}

}

case -1:

printf("Blad P3\n");

exit(1);

default:

wait(NULL);//wait dla P3

pid\_t pid3 = fork();

switch (pid3)

{

case 0:

/\* Potomny proces P4 \*/

printf("P4: id: %d id\_przodka: %d\n", getpid(), getppid());//trzeci potomek P1

exit(0);

case -1:

printf("Blad P4\n");

exit(1);

default:

wait(NULL); // Czekaj na zakończenie potomka P4

pid\_t pid4 = fork();

switch (pid4)

{

case 0:

/\* Potomny proces P5 \*/

printf("P5: id: %d id\_przodka: %d\n", getpid(), getppid());//czwarty potomek P1

exit(0);

case -1:

printf("Blad P5\n");

exit(1);

default:

wait(NULL);//wait dla P5

wait(NULL);//wait dla P2

exit(0);

}

}

}

}

return 0;

}

