Name: Kaustav Ghosh

Reg no: 180905188

Roll no: 29

Date:11/12/2020

Q1

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

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

{

int status;

pid\_t pid;

pid = fork();

if (pid == -1)

printf("\n ERROR child not created \n");

else if (pid == 0)

{

printf(" I am the child ! \n");

exit(0);

}

else

{

wait(&status);

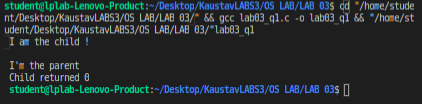
printf(" \n I'm the parent ");

printf(" \n Child returned %d\n ", status);

}

return 0;

}



Q2

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

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

{

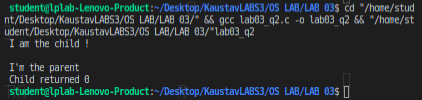
char \*cmdargs[] = {"./lab03\_q1", NULL};

execv(cmdargs[0], cmdargs);

printf("\n Wont be displayed \n");

return 0;

}



Q3

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

void display\_info()

{

//getpid() - will return process id of calling function

pid\_t process\_id = getpid();

//getppid() - will return process id of parent function

pid\_t p\_process\_id = getppid();

//printing the process ids

printf("The process id: %d\n", process\_id);

printf("The process id of parent function: %d\n", p\_process\_id);

}

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

{

int status;

pid\_t pid;

pid = fork();

if (pid == -1)

printf("\n ERROR child not created \n");

else if (pid == 0)

{

printf(" I am the child ! \n");

display\_info();

exit(0);

}

else

{

wait(&status);

printf(" \n I'm the parent ");

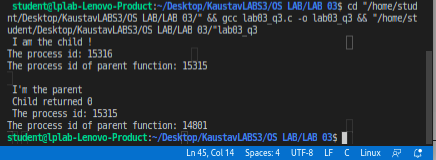
printf(" \n Child returned %d\n ", status);

display\_info();

}

return 0;

}



Q4 ZOMBIE

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

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

{

int status;

pid\_t pid;

pid = fork();

if (pid == -1)

printf("\n ERROR child not created \n");

else if (pid == 0)

{

printf(" I am the child ! \n");

exit(0);

}

else

{

// wait(&status);

printf(" \n I'm the parent ");

printf(" \n Child returned %d\n ", status);

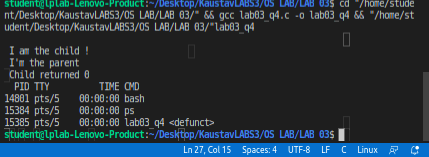
execlp("/bin/ps", "ps", NULL);

exit(0);

}

return 0;

}



Q5 ORPHAN

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

#include <stdio.h>

#include <stdlib.h>

void display\_info()

{

//getpid() - will return process id of calling function

pid\_t process\_id = getpid();

//getppid() - will return process id of parent function

pid\_t p\_process\_id = getppid();

//printing the process ids

printf("\n");

printf("The process id: %d\n", process\_id);

printf("The process id of parent function: %d\n", p\_process\_id);

}

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

{

int status;

pid\_t pid;

pid = fork();

if (pid == -1)

printf("\n ERROR child not created \n");

else if (pid == 0)

{

printf("////////////// I am the child //////////////////! \n");

// wait(&status);

sleep(3);

display\_info();

execlp("/bin/ps", "ps", NULL);

exit(0);

}

else

{

printf(" \n ////////////// I'm the parent //////////////////");

printf(" \n Child returned %d\n ", status);

exit(0);

}

return 0;

}

