Name : Dev Adnani  
SID : 202212012

Subject : Systems Programming

Assignment : 6

**1:**

202212012\_Lab6\_1\_expo\_child

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

float myPow(float x, int n)

{

float pow = 1;

while (n > 0)

{

pow \*= x;

n--;

}

return pow;

}

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

{

if (argc < 3)

{

printf("Invalid Number of Arguments...\n");

exit(0);

}

float exp = 1, value = atof(argv[1]);

int n = atoi(argv[2]);

if (value >= 1 || n < 1)

{

printf("Invalid values are provided...\n");

exit(1);

}

for (int i = 1; i <= n; i++)

{

int fac = 1;

for (int j = 2; j <= i; j++)

fac \*= i;

exp += myPow(value, i) / fac;

}

printf("Child (PID=%d): For x = %s the first %s terms yield is %f\n", getpid(), argv[1], argv[2], exp);

exit(exp);

}

202212012\_Lab6\_1\_expo\_parent

#include <stdio.h>

#include <stdlib.h>

#include <sys/wait.h>

#include <unistd.h>

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

{

if (argc < 3)

{

printf("Invalid Number of Arguments...\n");

exit(0);

}

int pid = fork();

if (pid != 0) // parent

{

int status;

int childPid = wait(&status);

printf("Child with PID %d has exited with value of e=%d\n", childPid, WEXITSTATUS(status));

if WIFEXITED (status)

printf("Child with PID %d WIFEXITED is terminated normally\n", childPid);

else if WIFSIGNALED (status)

printf("Child with PID %d WIFSIGNALED is terminated by signal %d\n", childPid, WTERMSIG(status));

else

printf("Child with PID %d has abnormal terminated\n", childPid);

printf("Parent (PID = %d): done\n", getpid());

}

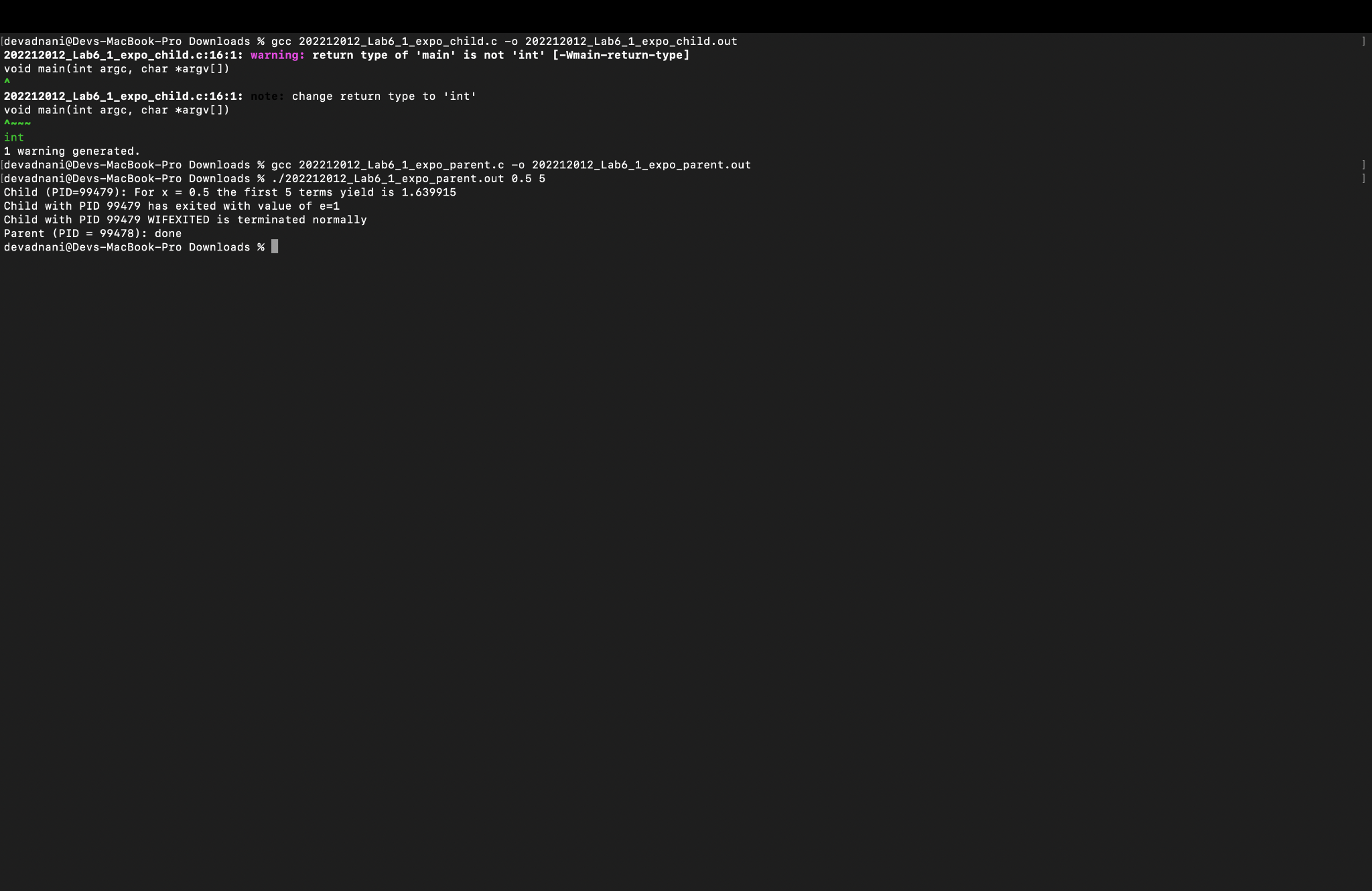
else

{

execl("./202212012\_Lab6\_1\_expo\_child.out", "./202212012\_Lab6\_1\_expo\_child.out", argv[1], argv[2], NULL);

}

}

Screenshot :   
  


**2:**

Note : We Have Used Same Child Code Which Was Used In Problem 1

202212012\_Lab6\_2\_expo\_parent

#include <stdio.h>

#include <stdlib.h>

#include <sys/wait.h>

#include <unistd.h>

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

{

if (argc < 3)

{

printf("Invalid Number of Arguments(Even no of arg required)...\n");

exit(0);

}

int p[argc];

for (int i = 0; i < argc - 1; i = i + 2)

{

p[i] = fork();

if (p[i] == 0)

{

execl("./202212012\_Lab6\_1\_expo\_child.out", "./202212012\_Lab6\_1\_expo\_child.out", argv[i + 1], argv[i + 2], NULL);

fprintf(stderr, "Could not execute the child process (PID=%d) \n", p[i]);

}

}

for (int i = 0; i < argc - 1; i += 2)

{

if (p[i] != 0)

{

int status;

int childPid = wait(&status);

printf("Child with PID %d has exited with value of e=%d\n", childPid, WEXITSTATUS(status));

if WIFEXITED (status)

printf("Child with PID %d WIFEXITED is terminated normally\n", childPid);

else if WIFSIGNALED (status)

printf("Child with PID %d WIFSIGNALED is terminated by signal %d\n", childPid, WTERMSIG(status));

else

printf("Child with PID %d has abnormal termination\n", childPid);

printf("Parent (PID = %d): done\n", getpid());

}

}

}

Screenshot :

