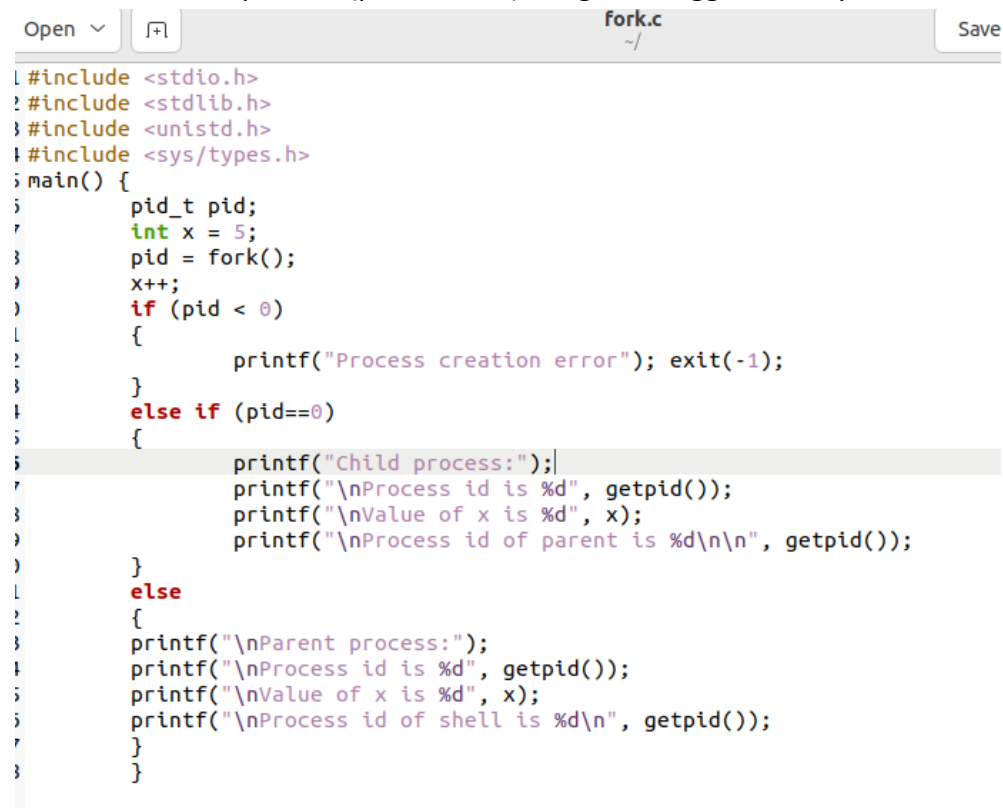


Nama : Ninda	Nilai
Nim : L200210265	
Nama dosen : Heru Setiya Nugraha,ST, M.Kom	Tanda tangan
Kelas : Praktikum Sistem Operasi E	

## Modul 8 System Call

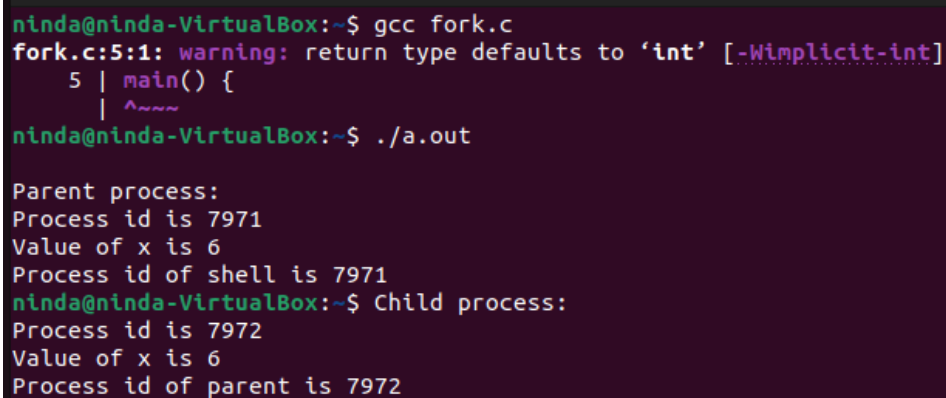
1. Membuat sebuah Child process (proses baru) dengan menggunakan system call 'fork'



```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 main() {
6     pid_t pid;
7     int x = 5;
8     pid = fork();
9     x++;
10    if (pid < 0)
11    {
12        printf("Process creation error"); exit(-1);
13    }
14    else if (pid==0)
15    {
16        printf("Child process:");
17        printf("\nProcess id is %d", getpid());
18        printf("\nValue of x is %d", x);
19        printf("\nProcess id of parent is %d\n\n", getpid());
20    }
21    else
22    {
23        printf("\nParent process:");
24        printf("\nProcess id is %d", getpid());
25        printf("\nValue of x is %d", x);
26        printf("\nProcess id of shell is %d\n", getpid());
27    }
28 }

```



```

ninda@ninda-VirtualBox:~$ gcc fork.c
fork.c:5:1: warning: return type defaults to 'int' [-Wimplicit-int]
5 | main() {
  | ^~~~~~
ninda@ninda-VirtualBox:~$ ./a.out

Parent process:
Process id is 7971
Value of x is 6
Process id of shell is 7971
ninda@ninda-VirtualBox:~$ Child process:
Process id is 7972
Value of x is 6
Process id of parent is 7972

```

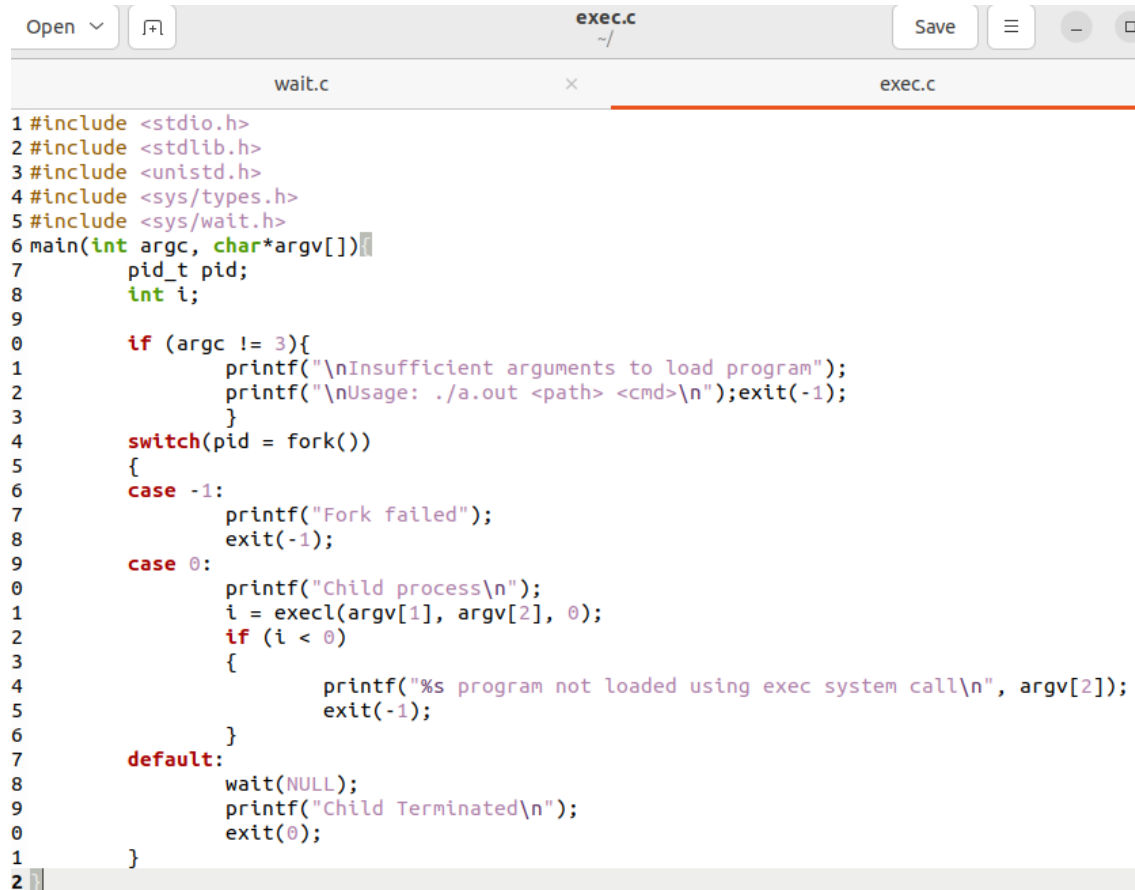
2. Menghentikan sementara(block) proses parent sampai dengan proses child selesai, menggunakan perintah system call "wait"

```
Open  wait.c  Sa
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 #include <sys/wait.h>
6 main(){
7     int i, status;
8     pid_t pid;
9     pid = fork();
10    if (pid < 0){
11        printf("\nPembuatan proses gagal\n");
12        exit(-1);
13    }
14    else if (pid > 0){
15        wait(NULL);
16        printf("\nParent starts\nNomor Genap :");
17        for (i=2;i<=10;i+=2)
18            printf("%3d",i);
19        printf("\nParent ends\n");
20    }
21    else if (pid == 0){
22        printf("Child starts\nNomor Ganjil:");
23        for (i=1;i<=10;i+=2)
24            printf("%3d",i);
25        printf("\nChild ends\n");
26    }
27 }
```

```
ninda@ninda-VirtualBox: ~
ninda@ninda-VirtualBox:~$ gcc wait.c
wait.c:6:1: warning: return type defaults to 'int' [-Wimplicit-int]
6 | main(){
  | ^~~~~
ninda@ninda-VirtualBox:~$ ./a.out
Child starts
Nomor Ganjil:  1  3  5  7  9
Child ends

Parent starts
Nomor Genap :  2  4  6  8 10
Parent ends
ninda@ninda-VirtualBox:~$
```

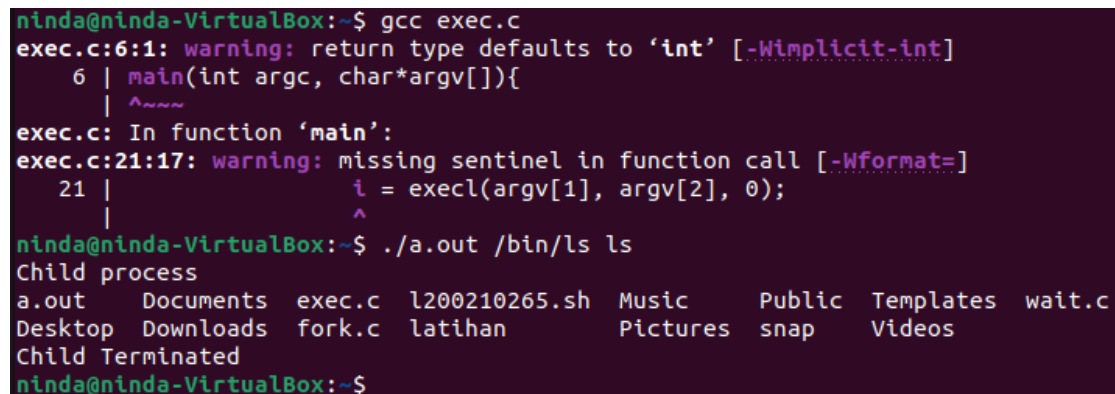
3. Loading program yang dapat dieksekusi dalam sebuah 'child' proses menggunakan perintah system call 'exec'



```
Open  [+]  exec.c  Save  [Menu]  [Close]

wait.c  x  exec.c

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 #include <sys/types.h>
5 #include <sys/wait.h>
6 main(int argc, char*argv[])
7     pid_t pid;
8     int i;
9
10    if (argc != 3){
11        printf("\nInsufficient arguments to load program");
12        printf("\nUsage: ./a.out <path> <cmd>\n");exit(-1);
13    }
14    switch(pid = fork())
15    {
16    case -1:
17        printf("Fork failed");
18        exit(-1);
19    case 0:
20        printf("Child process\n");
21        i = execl(argv[1], argv[2], 0);
22        if (i < 0)
23        {
24            printf("%s program not loaded using exec system call\n", argv[2]);
25            exit(-1);
26        }
27    default:
28        wait(NULL);
29        printf("Child Terminated\n");
30        exit(0);
31    }
32 }
```



```
ninda@ninda-VirtualBox:~$ gcc exec.c
exec.c:6:1: warning: return type defaults to 'int' [-Wimplicit-int]
   6 | main(int argc, char*argv[]){
     | ^~~~~
exec.c: In function 'main':
exec.c:21:17: warning: missing sentinel in function call [-Wformat=]
   21 |             i = execl(argv[1], argv[2], 0);
       |             ^
ninda@ninda-VirtualBox:~$ ./a.out /bin/ls ls
Child process
a.out  Documents  exec.c  l200210265.sh  Music  Public  Templates  wait.c
Desktop Downloads fork.c latihan  Pictures  snap  Videos
Child Terminated
ninda@ninda-VirtualBox:~$
```

#### 4. Menampilkan status file menggunakan perintah system call 'stat'

```
stat.c
~/
Open ▾ [?]

1 #include <stdio.h>
2 #include <sys/stat.h>
3 #include <stdlib.h>
4 #include <time.h>
5 int main(int argc, char*argv[])
6 {
7     struct stat
8     file; int n;
9     if (argc != 2)
10    {
11        printf("Usage: ./a.out <filename>\n"); exit(-1);
12    }
13    if ((n = stat(argv[1], &file)) == -1)
14    {
15        perror(argv[1]);
16        exit(-1);
17    }
18    printf("User id : %d\n", file.st_uid);
19    printf("Group id : %d\n", file.st_gid);
20    printf("Block size : %d\n", file.st_blksize);
21    printf("Blocks allocated : %d\n", file.st_blocks);
22    printf("Inode no. : %d\n", file.st_ino);
23    printf("Last accessed : %s", ctime(&(file.st_atime)));
24    printf("Last modified : %s", ctime(&(file.st_mtime)));
25    printf("File size : %d bytes\n", file.st_size);
26    printf("No. of links : %d\n", file.st_nlink);
27    printf("Permissions :");
28    printf( (S_ISDIR(file.st_mode)) ? "d" : "-");
29    printf( (file.st_mode & S_IRUSR) ? "r" : "-");
30    printf( (file.st_mode & S_IWUSR) ? "w" : "-");
31    printf( (file.st_mode & S_IXUSR) ? "x" : "-");
32    printf( (file.st_mode & S_IRGRP) ? "r" : "-");
33    printf( (file.st_mode & S_IWGRP) ? "w" : "-");
34    printf( (file.st_mode & S_IXGRP) ? "x" : "-");
35    printf( (file.st_mode & S_IROTH) ? "r" : "-");
36    printf( (file.st_mode & S_IWOTH) ? "w" : "-");
37    printf( (file.st_mode & S_IXOTH) ? "x" : "-");
38    printf("\n");
39    if (file.st_mode & S_IFREG)
40        printf("File & type : Regular\n");
41    if (file.st_mode & S_IFDIR)
42        printf("File & type : Directory\n");
43    }
```

```
ninda@ninda-VirtualBox:~$ ./a.out
Usage: ./a.out <filename>
ninda@ninda-VirtualBox:~$ ./a.out stat.c
User id : 1000
Group id : 1000
Block size : 4096
Blocks allocated : 8
Inode no. : 523295
Last accessed : Wed Dec  7 03:54:25 2022
Last modified : Wed Dec  7 03:54:16 2022
File size : 1385 bytes
No. of links : 1
Permissions :-rw-rw-r--
File & type : Regular
ninda@ninda-VirtualBox:~$
```

5. Menampilkan isi direktori menggunakan perintah system call 'reader'

```
Open ▾  dirlist.c ~/
1 #include <stdio.h>
2 #include <dirent.h>
3 #include <stdlib.h>
4 main (int argc, char *argv[])
5     struct dirent *dptr;
6     DIR *dname;
7
8     if (argc !=2)
9     {
10         printf("Usage : ./a.out <dirname>\n");
11         exit(-1);
12     }
13     if ((dname = opendir(argv[1])) == NULL)
14     {
15         perror(argv[1]);
16         exit(-1);
17     }
18     while (dptr=readdir(dname))
19         printf("%s\n", dptr->d_name);
20
21     closedir(dname);
22
```

```
ninda@ninda-VirtualBox: ~
ninda@ninda-VirtualBox:~$ gcc dirlist.c
dirlist.c:4:1: warning: return type defaults to 'int' [-Wimplicit-int]
4 | main (int argc, char *argv[])
  | ~~~~
ninda@ninda-VirtualBox:~$ ./a.out praktikum
..
.
txt.txt
ninda@ninda-VirtualBox:~$
```