Q1) close(fd) returns 0 if the fd is already closed.

True

False

Q2)

main()

{

int pid1,pid2,pid3,pid4;

pid1=fork();

pid2=fork();

pid3=fork();

pid4=fork();

if(pid1==0||pid2==0||pid3==0||pid4==0){

exit(0);

}

else

{

for(;;);

} }

How many zombie processes are created?

4

16

12

None of the options

Q3)

//Note: Only the operational part of the program is shown below.

int fd=open("sample.txt",O\_CREAT|O\_RDWR);

int long n=lseek(fd,20,SEEK\_END);

char buff1[15];

for(int i=0;i<15;i++)

{

buff1[i]='A';

}

n=write(fd,buff1,15);

close(fd);

sample.txt would contain?

a. None of the options

b. 35 Characters

c. 15 consecutive 'A' characters only

d. 20 NULL characters only

Q4)

//Note: Only the operational part of the program is shown below.

int pid = fork();

if(pid==0)

{

exit(0);

}

else if(pid<0)

{

printf("Error");

}

else

{

while(1)

sleep(1);

}

The above code segment would lead to:

a. A zombie process

b. An orphan process

c. WIFSIGNALED(status) =TRUE

d. None of the options

Q5)

void main()

{

fork();

fork();

fork()

}

The number of processes that will not fork a child process would be 4

True

False

Q6

Any thread can kill any other thread within the program

• TRUE

• FALSE

Q7

main(){

int num1,num2;

int fd1=open("input.txt",O\_RDONLY);

int fd2=open("output.txt",O\_RDWR);

printf("\nSUNDAY");

printf("\nMONDAY");

int c1=dup(0);

dup2(fd1,0);

scanf("%d",&num1);

scanf("%d",&num2);

printf("\nTUESDAY");

int c2=dup(1);

dup2(fd2,1);

printf("\nThe sum of two numbers is %d\n",num1+num2);

dup2(1,c2);

dup2(0,c1);

printf("\nWEDNESDAY");

printf("\nTHURSDAY");

}

What is the output of the above program?

• SUNDAY

MONDAY

TUESDAY

• TUESDAY

• WEDNESDAY

THURSDAY

• SUNDAY

MONDAY

Q8

void handler(){

exit(0);

}

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

int i, pid;

signal(SIGINT, handler);

for(int i=1;i<4;i++)

fork();

if((pid=fork()) == 0)  {

setpgid(0, getpid());

for(;;){

sleep(1);

}}

else

{

for(;;){

sleep(1);

}}

}

How many processes will not be killed when you run the program and you press ctrl-c ?

• 2

• 8

• 4

• None of the options

Q9

main() {

int a, b;

long int count;

char \*buffer;

f1=open("sample.txt", O\_RDONLY);

dup2(f1, STDIN\_FILENO);

c=lseek(STDIN\_FILENO, 0, SEEK\_END);

lseek(STDIN\_FILENO, 0, SEEK\_SET);

buffer=malloc(count);

read(0, buffer, count);

write(STDOUT\_FILENO, buffer, count);

}

What does the above program do?

• Copies contents from keyboard to a file

• Displays the contents of a file on the monitor/screen.

• None of the options

• Copies files

Q10

main()

{

int fd3=open("sample.txt",O\_RDONLY);

char buffer[95];

long int n;

for(int i=0;i<15;i++)

{

n+=read(fd3,buffer,15);

}

for(int i=0;i<5;i++){

printf("\n%d", (int) lseek(fd3, -100, SEEK\_END));

}

close(fd3);

}

Assuming that the size of sample.txt is 50 bytes, what would be the output of the final printf

statement in this program?

• -100

• -450

• -50

• None of the options

Q11

The number of forked processes from the main process with immediate descendants is 3 (three)

main(){

fork();

fork();

fork();

}

TRUE

FALSE

Q12

The address of the first element of an array of 10 integers is a constant pointer.

TRUE

FALSE

Q13

int (\*f1)(int,int);

int (\*f2)(int,int);

int a1(int a,int b) {

if(a<=b)return a;

else return b;

}

int a2(int a,int b) {

if(a>=b)return a;

else return b;

}

int calculate(int (\*f1)(int,int), int (\*f2)(int,int), int

p1,int p2,int p3,int p4) {

int k=f1(p1,p2);

int j=f2(p3,p4);

if(k>=j)return k;

else return j;

return k;

}

Which of the following function calls would

return maximum( minimum(20,30) , maximum(25,15) ) ?

int ret=calculate(f1,f2,20,30,25,15)

int ret=calculate(&a1,&a2,20,30,25,15)

int ret=calculate(a1,a2,20,30,25,15)

None of the above

Q14  
int func1(int x){  
return(x\*x + x - 6);  
}  
int func2(int x){  
return(x\*x - 3\*x +1);  
}  
int main(){  
int (\*pt)(int)=func2;  
printf("%d\n", pt(3));  
pt=func1;  
int ret=pt(2);  
printf("%d\n", ret));  
exit(0);  
}  
What would be the value of ret?  
· 0  
· 1  
· 0  
· None of the above