Rabia batool

2022-BSE-064

Group#B

Operating System

Lab#09

Submitted to Sir Shehzad

Practice program:

```
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;

shmid=shmget((key_t)2345, 1024, 0666|IPC_CREAT);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);
```

return 0;

```
 GNU nano 6.2
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;

shmid=shmget((key_t)2345, 1024, 0666|IPC_CREAT);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);
return  0;
```

```
~$ gcc sm.c
~$ ./a.out
Key of shared memory is 0
Process attached at 0x7f15ebb41000
Enter some data to write to shared memory
hafsa asad
You wrote : hafsa asad
```

Practice program 2:

GNU nano 6.2 smh.c

#include<stdio.h>

#include<stdlib.h>

```
#include<unistd.h>


#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;
shmid=shmget((key_t)2345, 1024, 0666);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Data read from shared memory is : %s\n",(char
*)shared_memory);
}
```

```
 GNU nano 6.2
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>

#include<sys/shm.h>
#include<string.h>
int main()
{
int i;
void *shared_memory;
char buff[100];
int shmid;
shmid=shmget((key_t)2345, 1024, 0666);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Data read from shared memory is : %s\n",(char
*)shared_memory);
}
```

```
~$ touch smh.c
~$ nano smh.c
~$ gcc smh.c
~$ ./a.out
Key of shared memory is 0
Process attached at 0x7f25e9cbb000
Data read from shared memory is : hafsa asad

~$
```

VIVA QS:

Q1. What does shmget() function return on success?

On success, the `shmget()` function returns a non-negative integer, which is a shared memory identifier (shmid). This identifier is unique within the system and can be used by other processes to access the shared memory segment.

Q2. What does shmat() function return on success?

On success, the `shmat()` function returns the address of the attached shared memory segment in the calling process's address space. This is a pointer that can be used to access the shared memory directly.

Q3. The value of segment size in shmget() is a round-up value to a multiple of ____.

The value of segment size in `shmget()` is a round-up value to a multiple of the system page size.2^n

Q4. Can a process create two shared segment? Will there returned identifiers be same or

different?

Yes, a process can create two shared segments. In fact, a process can create any nUmber of shared segments, as long as it has sufficient resources and permissions.The identifiers returned by `shmget()` for each shared segment will be different. This is because each shared segment is unique and has its own control structure in the kernel. The identifier acts as a reference to the shared segment and is used by other processes to access it

Q5. Which function can be used to detach the shared segment from the address space of the process?

The function used to detach a shared segment from the address space of the process is shmdt().

Task 1

```
#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/shm.h>

#include<string.h>

int main()

{pid_t p;

int i;

void *shared_memory;

char buff[100];

int shmid;

p=fork();

if(p==0)

{
```

```
int i;
void *shared_memory;
char buff[100];
int shmid;
printf("this is child \n");
shmid=shmget((key_t)2345, 1024, 0666);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Data read from shared memory is : %s\n",(char
*)shared_memory); }
else{
printf("this is parent  \n");
shmid=shmget((key_t)2048, 1024, 0666|IPC_CREAT);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);}
return  0;
}
```

```
 GNU nano 6.2
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/shm.h>
#include<string.h>
int main()
{pid_t p;
int i;
void *shared_memory;
char buff[100];
int shmid;
p=fork();
if(p==0)
{
int i;
void *shared_memory;
char buff[100];
int shmid;
printf("this is child \n");
shmid=shmget((key_t)2345, 1024, 0666);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Data read from shared memory is : %s\n",(char
*)shared_memory); }

else{
printf("this is parent  \n");
shmid=shmget((key_t)2048, 1024, 0666|IPC_CREAT);
printf("Key of shared memory is %d\n",shmid);
shared_memory=shmat(shmid,NULL,0);
printf("Process attached at %p\n",shared_memory);
printf("Enter some data to write to shared memory\n");
read(0,buff,100);
strcpy(shared_memory,buff);
printf("You wrote : %s\n",(char *)shared_memory);}
return  0;
}
```

```
~$ nano lab.c
~$ gcc lab.c
~$ ./a.out
this is parent
Key of shared memory is 0
this is child
Process attached at 0x7f59499bb000
Enter some data to write to shared memory
Key of shared memory is -1
Process attached at 0xffffffffffffffff
this is lab 9
You wrote : this is lab 9
```