Siganl _Handling (https://github.com/amalpoulose/system-programming-signal-handling)

---

1. Write a program to deliver the alarm signal periodically after (n-1)sec from the last interrupt.

Hint : Start first alarm at 10 sec then next alarm at 9sec so on up to n=0 terminate .

```
#include<stdio.h>
#include<signal.h>
#include<unistd.h>
int seconds=10;
void my_isr(int n)
{
        printf("Alarm clock\n");
       if(seconds==2)
                signal(SIGALRM,SIG_DFL);
        seconds--;
        printf("Setting alarm for %d second\n",seconds);
        alarm(seconds);
}
int main(void)
{
        signal(SIGALRM,my_isr);
        printf("Setting alarm for %d second\n",seconds);
        alarm(seconds);
        while(1);
         return 0;
}
```

2. Create a watch dog timer in parent which should watch T.A.T of its child and terminate the child.

Condition: The child have random delay(1-10 sec)

If the child take more than 5 sec then parent terminate it.

Hint : fork(),sleep?(),kill(),alarm()

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/wait.h>
#include<unistd.h>
#include<sys/types.h>
#include<signal.h>
int main(void)
{
        int r_value,cid;
        srand(getpid());
        r_value=rand()%10+1;
      printf("Delay = %d\n",r_value);
        if((cid=fork())==0)
        {
        printf("child starts.....\n");
        sleep(r_value);
        printf("child ends\n");
        exit(0);
        }
        else
        {
        alarm(6);
```

```
        void my_isr(int n)
        {
                kill(cid,9);
                  printf("Child Terminated Forcefully\n");
        }
        signal(SIGALRM,my_isr);
        wait(0);
        signal(SIGALRM,SIG_IGN);
        }
        return 0;
}
```

3. Create a function named “find-isr”,which when calling by passing an interrupt number then it should inform the action of that signal in current program.

Signal action----□defaulted

o/p->Ignore

userdefine.

Hint : use signal( ) function to return value.

```
#include<stdio.h>
#include<signal.h>
int main(int argc , char **argv)
{
        if(argc!=2)
        {
        printf("Usage : ./a.out sig_no\n");
        return;
        }
        int sig_no;
        void (*action)(int);
        sig_no=atoi(argv[1]);
        action=signal(sig_no,SIG_IGN);
        signal(sig_no,action);
        if(action==SIG_DFL)
        printf("Terminate\n");
        else if(action==SIG_IGN)
        printf("Ignore\n");
        else
        printf("User defined\n");
        return 0;
}
```

4. Write a program to remove the zombie.

Hint : Use SIGCHLD &signal( ).

```
#include<signal.h>
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<unistd.h>
int pid;
void func(int n)
{
wait(0);
printf("Zombie of Id %d is removed\n",pid);
}
int main(void)
{
signal(SIGCHLD,func);
printf("program start.....\n");
if((pid=fork())==0)
{
 sleep(10);
 printf("child\n");
}
else
{
printf("parent continue....\n");
while(1);
}
return 0;
}
```

5. Write a program to ignore the termination of process when its terminal will close.

Hint : use SIGHUP signal & signal().

```
#include<signal.h>

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

int main(void)

{


signal(SIGHUP,SIG_IGN);

printf("program start.....\n");

while(1);

return 0;

}
```

6. Write a program to install ISR(handler) for SIGINT and SIGQUIT . Restore the SIGINT to default after 3 times occurrence.SIGQUIT to default after 5 times occurrence.

```
#include<signal.h>

#include<stdio.h>


void func1(int);

void func2(int);


int count1=1;

int count2=1;


int main(void)

{

signal(SIGINT,func1);

signal(SIGQUIT,func2);

printf("Program started.....\n");

while(1);

return 0;

}


void func1(int n)

{

if(count1==3)

signal(SIGINT,SIG_DFL);

count1++;

}
```

```
void func2(int n)
{
if(count2==5)
signal(SIGQUIT,SIG_DFL);
count2++;
}
```

7. Write a program to create 3 child process from common parent use random delay between 1 to 10 sec in each child. Use alarm() in parent in such a manner that

Child1 should not exceeds more than 4 sec.

Child2 should not exceeds more than 6 sec.

Child3 should not exceeds more than 8 sec.

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/wait.h>
#include<unistd.h>
#include<sys/types.h>
#include<signal.h>
int main(void)
{
        int r_value,cid[3],i,delay=4,temp,s;
        srand(getpid());
        r_value=rand()%10+1;
        printf("Delay = %d\n",r_value);
        for(i=0;i<3;i++)
        {
        if((cid[i]=fork())==0)
        {
                printf("child %d starts.....\n",i+1);
                sleep(r_value);
                printf("child %d ends\n",i+1);
                exit(0);
        }
        else;
        }
```

```
        i=0;
          void my_isr(int n)
          {
          kill(cid[i],9);
          printf("Child %d Terminated Forcefully\n",i+1);
         }
        alarm(delay+1); // in case delay = 4
        while(i<3)
       {
        temp=0;
        signal(SIGALRM,my_isr);
        wait(0);
        signal(SIGALRM,SIG_IGN);
        if(delay-r_value>0)
            temp=delay-r_value;
        delay+=2;
        i++;
        alarm(2+temp);
       }
return 0;
}
```

8.Write a Program parent has to get the data from user and store the data in file.after that parent process will send one signal to child process.child process after receving the signal,open the file and convert data to oppsite case and store again in same file.

Hint : Use pause() in child process.

Use signal num 10(usersignal) in parent.

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<signal.h>
#include<sys/wait.h>
int main(void)
{
        int cid;
        FILE *fp;
        if((cid=fork())==0)
        {
        void my_isr(int n)
        {
                char ch;
                fp=fopen("filen","r+");
                while((ch=fgetc(fp))!=EOF)
                {
                        if((ch>64 && ch<91)||(ch>96 && ch<123))
                        {
                                ch=ch^32;
                                fseek(fp,-1,SEEK_CUR);
                                fputc(ch,fp);
                        }
```

```
            }
            fclose(fp);
            printf("Done\n");
            exit(0);
        }
        signal(10,my_isr);
        pause();
        }
        else
        {
        fp=fopen("filen","w");
        char s[50];
        printf("enter String : ");
        scanf("%[^\n]",s);
        fputs(s,fp);
        fclose(fp);
        kill(cid,10);
        wait(0);
        }
        return 0;
}
```