Topic :

Page : 3

Date : 1 / 8 / 19

Write a Program to check whether your marks 1st, 2nd or 3rd Division.

```
#include <stdio.h>
int main()
{
int m;
printf("enter your marks\n");
scanf("%d", &m);
if (m>=90 && m<100)
printf(" first division\n");
else if (m>=80 && m<90)
printf(" second division\n");
else printf("third division\n");
}
```

```
[student1@Server-lab7 ~]$ gcc -o nishigrade nishigrade.c
[student1@Server-lab7 ~]$ ./nishigrade
enter your marks
91
first division
[student1@Server-lab7 ~]$
```

Topic :

PIONEER
Page : 4
Date : 1 / 8 / 19

WAP to display the factorial of a given number.

```
#include<stdio.h>
int main()
{
 int n;
 printf ("enter the number for factorial \n");
 scanf ("%d", &n);
 int i, fact = 1;
 for( i=1 ; i<=n ; i++)
{
   fact = fact * i ;
 }
 printf (" factorial of %d is %d \n", n, fact);
}
```

```
[student1@Server-lab7 v]$ gcc -o nishifact nishifact.c
[student1@Server-lab7 v]$ ./nishifact
enter the number for factorial
5
factorial of 5 is 120
[student1@Server-lab7 v]$
```

Topic :

PIONEER
Page : 5
Date : 1 / 8 / 19

WAP to generate the fibbonacci series for n number of terms

```
#include<stdio.h>
int main()
{
 int f=0, s=1, n, i, sum=0;
 printf("enter the number of terms\n");
 scanf("%d", &n);
 printf("fibonacci series is\n");
 printf("%d\n%d\n", f, s);
 for(i=2; i<n; i++)
{
   sum = f+s;
   f=s;
   s= sum;
   printf("%d\n", s);
}
}
```

```
[student1@server-lab7 ~]$ gcc -o nishifibo nishifibo.c
[student1@server-lab7 ~]$ ./nishifibo
enter the number of terms
5
fibonacci series is:
0
1
1
2
3
[student1@server-lab7 ~]$
```

Topic :

PIONEER

Page : 6

Date : 1 / 8 / 19

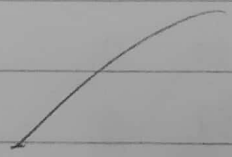
WAP to check whether a given number is Prime or Not.

```
#include<stdio.h>
int main()
{
  int n;
  printf("enter the number\n");
  scanf("%d", &n);
  int c=0, i;
  for(i=1; i*i<=n; i++)
{
  if(n%i==0)
   c++;
}
 if(c==1)
 printf("prime number\n");
else printf("not a prime number");
}
```

```
[student1@Server-lab7 ~]$ gcc -o nishiprime nishiprime.c
[student1@Server-lab7 ~]$ ./nishiprime
enter the number
7
prime number
[student1@Server-lab7 ~]$
```

WAP to check whether a given number is even or odd.

```
#include <stdio.h>
int main()
{
  int n;
  printf ("enter a number \n");
  scanf ("%d", &n);
  if (n%2==0)
  printf ("even number \n");
  else printf ("odd number \n");
}
```



```
[student1@Server-lab7 ~]$ gcc -o nishievenodd nishievenodd.c
[Student1@Server-lab7 ~]$ ./nishievenodd
enter a number
5
odd number
[student1@Server-lab7 ~]$.
```

Page 8
Date 1/8/19

WAP to print the Process ID and Parent Process ID of a program and execute system ("ps") and system ("ps-al") functions.

```
#include<stdio.h>
#include<sys/types.h>
void main()
{
   system("ps");
   printf("process id = %d", getpid());
   printf("parent process id = %d", getppid());
   system("ps-al");
}
```

08/09/19

```
[student1@server -lab7 ~]$ gcc -o nishiprocess nishiprocess.c
[student1@server -lab7 ~]$ ./nishiprocess
  PID        TTY          TIME      CMD
 2406        pts/2        00:00:00  bash
 5039        pts/2        00:00:00  nishiprocess
 5040        pts/2        00:00:00  ps
```

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
0	T	794	3919	3371	0	80	0	-	1428	signal	pts/10	00:00:00	vim
0	T	794	4064	3316	0	80	0	-	3100	signal	pts/5	00:00:00	vim
0	T	794	4254	3867	0	80	0	-	3061	signal	pts/20	00:00:00	vim
0	T	794	4480	3485	0	80	0	-	3093	signal	pts/8	00:00:00	vim
0	S	793	4662	4292	0	80	0	-	3095	poll_s	pts/3	00:00:00	vim
0	S	794	4701	4713	0	80	0	-	3093	poll_s	pts/20	00:00:00	vim
0	S	794	4875	2530	0	80	0	-	3095	poll_s	pts/10	00:00:00	vim
0	S	794	4919	4815	0	80	0	-	3095	signal	pts/7	00:00:00	vim
0	S	794	4979	3867	0	80	0	-	3095	poll_s	pts/21	00:00:00	vim
0	S	794	4990	2225	0	80	0	-	3095	poll_s	pts/15	00:00:00	vim
0	S	794	4992	4991	0	80	0	-	3093	poll_s	pts/2	00:00:00	vim
0	S	794	5039	2406	0	80	0	-	477	wait	pts/2	00:00:00	nishi process
0	R	794	5041	5039	0	80	0	-	1188	-	pts/2	00:00:00	ps
0	S	794	18700	4575	0	80	0	-	3100	poll_s	pts/14	00:00:00	vim

process id = 5039 parent process id = 2406 [student1@server -lab7 ~]$

Topic :

PIONEER
Page : 9
Date : 08/ 8 / 19

ASSIGNMENT- 1a

Create a child process during the runtime of source code.

1. check the PID of child process.
2. Who is the parent of your child process and figure out its PID?
3. Why this PPID is not changing where as the PID of your process is changing every time?

```
#include <stdio.h>
# include <stdlib.h>
# include <sys/types.h>
# include <unistd.h>
pid_t fork(void);
int main()
{
 pid_t p;
 char *msg;
 int n, i;
 printf("the fork program starting \n");
 p= fork();
 switch(p)
{
 case -1:
```

```
[student@server -lab7 ~] $ gcc -o hi Assign1a-70.c
[student@server -lab7 ~] $ ./hi

The fork program is starting
The parent process id = 2778
This is the parent
The child process id = 2779
This is the child
This is the parent
This is the child
This is the parent
This is the child
2005

This is the child
  PID    TTY      TIME     CMD
 2005    pts/17   00:00:00 bash
 2778    pts/17   00:00:00 hi
 2779    pts/17   00:00:00 hi
 2781    pts/17   00:00:00 ps
```

```
perror ("Fork failed");
exit(1);
case 0:
msg = "this is the child";
n=5;
printf ("the child process id = %d \n", getpid());
break;
default:
msg = "this is the parent";
n=3;
printf ("the parent process id = %d\n", getppid());
break;
}

for ( ;n>0;n--)
{

puts(msg);
sleep(1);
}

printf ("%d\n", getppid());
system ("ps");
exit(0);
}
```

The PPID is not changing because bash is running at the background whereas the process is running at the foreground. Hence the PID is changing whereas the PPID is not changing.

[student@server-lab7 ~] $ This is the child

PID	TTY	TIME	CMD
2605	pts/17	00:00:00	bash
2779	pts/17	00:00:00	hi
2782	pts/17	00:00:00	ps

Topic :

PIONEER

Page : 11
Date : 8/ 8 /19

ASSIGNMENT - 1b.

Edit your source code of assignment 1 in such a way so that-
1. Your child process becomes zombie
2. Show the zombie status
3. How could you make your child process orphan ?
4. Who is the parent of that orphan process?
5. check the status of orphan Process.

For Zombie Process:

```
#include <stdio.h>
#include< stdlib.h>
#include <sys/types.h>
#include <unistd.h>
pid_t fork(void);
int main()
{
 pid-t p;
 char *msg;
 int i,n;
 printf (" The Fork Program Starting \n");
 p = fork();
 switch(p)
```

```
[student1@ Server -lab7 ~] $ gcc -o lb-70 Assignment1b_70.c
[student1@ Server- lab7 ~] $ ./lb-70

The fork program is starting
The parent process id =3016
This is the parent
The child process id = 3017
This is the child
This is the parent
This is the child
This is the parent
This is the child
This is the parent
3016

  PID    TTY      TIME     CMD
 1626   pts/5   00:00:00   bash
 3016   pts/5   00:00:00   122
 3017   pts/5   00:00:00   122
 3018   pts/5   00:00:00   ps
```

```
{
 case -1:
 perror ("Fork Failed");
 exit(1);
 case 0:
 msg = "This is the child";
 n=3;
 printf ("The child Process ID = %d\n", getpid());
 break;
 default:
 msg = "This is the Parent";
 n=5;
 printf ("The Parent Process ID = %d\n", getppid());
 break;
}
for(; n>0 ; n--)
{
  puts(msg);
  sleep(1);
}
printf ("%d\n", getppid());
system ("ps");
exit(0);
}
```

This is the parent
1828

PID	TTY	TIME	CMD	
1828	pts/5	00:00:00	bash	
3016	pts/5	00:00:00	122	
3017	pts/5	00:00:00	122	<defunct>
3019	pts/5	00:00:00	ps	

For orphan Process

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/types.h>
#include<unistd.h>
pid_t fork(void);
int main()
{
  pid_t p;
  char *msg;
  int n, i;
  printf("The Fork Program Starting\n");
  p = fork();
  switch(p)
  {
    case -1:
    perror("Fork Failed");
    exit(1);
    case 0:
    msg="This is the child";
    n=5;
    printf("The child Process ID = %d\n", getpid());
    break;
    default:
```

```
    msg = "this is the parent";
    n=3;
    printf (" The Parent Process ID = %d\n", getppid());
    break;
  }
  for(; n>0; n--)
   {
     puts(msg);
     sleep(1);
   }
  printf("%d\n", getppid());
  system("ps");
  exit(0);
}
```

The Parent of the Orphan Process is init process which was called booting.

ASSIGNMENT - 1 OPTIONAL

Write a C program to resolve the problem of orphan process creation.
Use the system call wait(), so the parent process should wait till the termination of created child process.

```
#include <stdio.h>
#include <stdlib.h>
#include<sys/types.h>
#include<sys/wait.h>
#include<unistd.h>
pid_t wait (int *stat_loc);
pid_t fork (void);
int system (const char *string);
int main()
{
 pid_t pid;
 int n, exit_code;
 char *msg;
 printf ("The Fork Program is Running\n");
 pid = fork();
 switch(pid)
 {
```

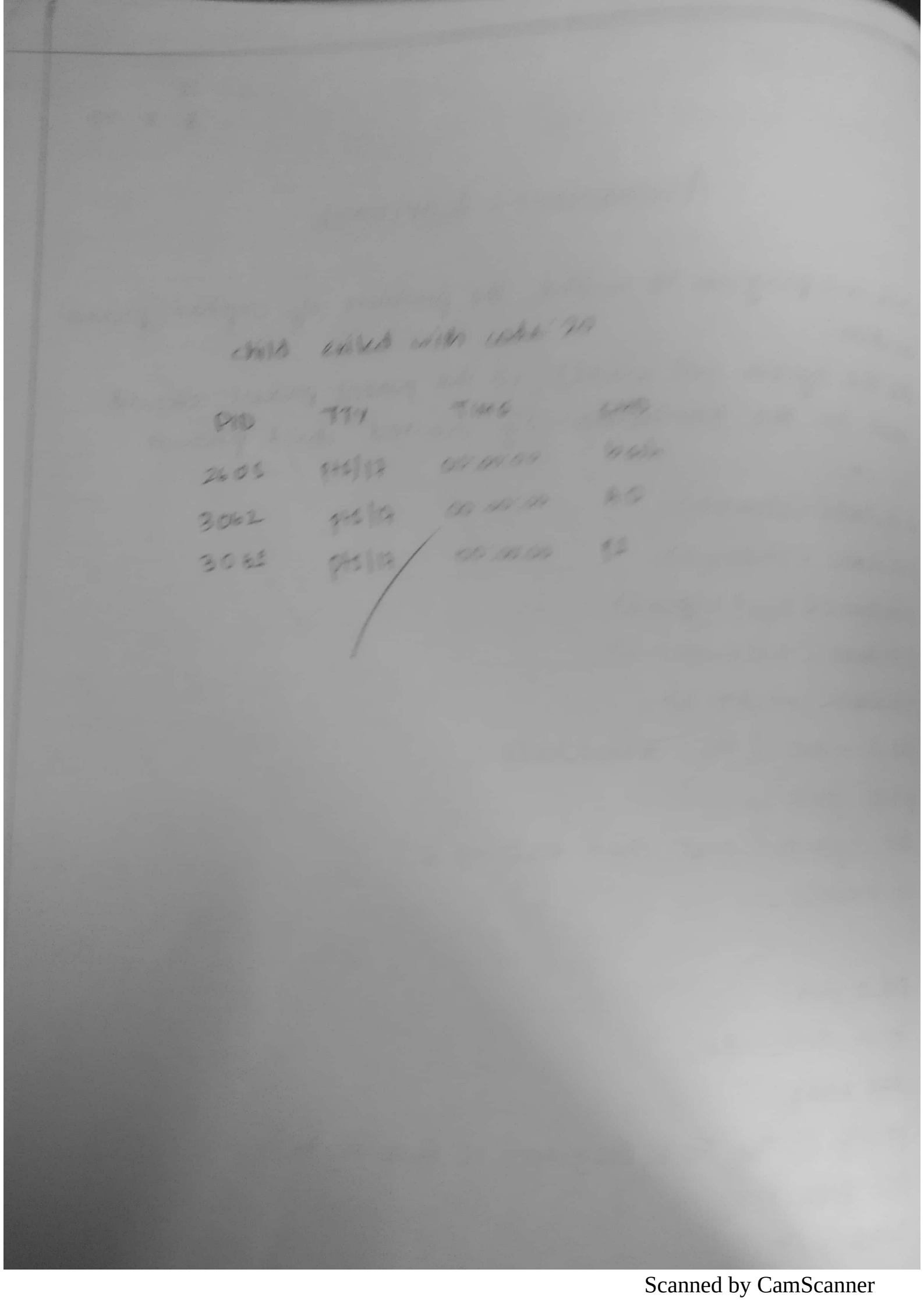
```
[student@server -lab7 ~] $ gcc -o A0 Assign0_70.c
[student@server -lab7 ~] $ ./A0

The fork program is running
The parent is running
The child process is running
The parent is running
The child process is running
The parent is running
The child process is running
The child process is running
The child process is running
```

PID	TTY	TIME	CMD
2605	pts/17	00:00:00	bash
3062	pts/17	00:00:00	A0
3063	pts/17	00:00:00	A0
3064	pts/17	00:00:00	ps

```
The child has finished PID = 3063
The bash process PID = 2605
The process id = 3062
```

```
        case -1:
        perror("The child Process is Running Failed \n");
        break;
        case 0:
        msg = "The child Process is Running";
        exit_code = 20;
        n=5;
        break;
        default:
        msg = "The Parent is Running";
        exit_code=0;
        n=3;
        Break;
    }
for( ; n>0 ; n--)
{
    puts(msg);
    sleep(1);
}
if (pid != 0)
{
  pid_t child_pid;
  int stat_val;
  child_pid = wait (& stat_val);
```



```
printf ("The child has finished PID = %d\n", child_pid);
printf ("The Bash Process PID = %d\n", getppid());
printf ("The Process ID = %d\n", getpid());
if (WIFEXITED (stat_val))
printf ("child exited with code: %d\n", WEXITSTATUS
                                        (stat_val));
else
printf ("child Terminated Abnormally");
}
system ("ps");
exit (exit_code);
}
```

Topic :

PIONEER

Page : 18

Date : 8 / 8 / 19

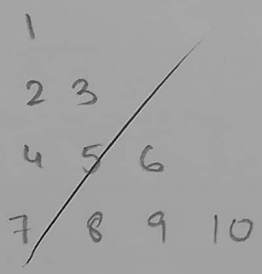
WAP to print the following pattern where the Number of lines is taken as input.

```
1
2 3
4 5 6
7 8 9 10
```

```
#include<stdio.h>
void main()
{
  int c=1,i,j,n;
  printf("Enter the Number of lines:");
  scanf("%d",&n);
  for(i=1;i<=n;i++)
  {
    for(j=1;j<=i;j++)
      printf("%d\t",c++);
    printf("\n");
  }
}
```

[student@server -lab7 ~]$ gcc -o p1 pattern_70.c
[student@server -lab7 ~]$ ./p1

enter the Number of Lines : 4



Topic :

PIONEER®

Page : 19

Date : / /

WAP to print the following pattern where the no. of lines is taken as input.

```
1
1 2
1 2 3
1 2 3 4
```

```
#include<stdio.h>
void main()
{
  int i,j,n;
  printf("Enter a Number of lines: ");
  scanf("%d",&n);
  for(i=1;i<=n;i++)
  {
    for(j=1;j<=i;j++)
      printf("%d\t",j);
    printf("\n");
  }
}
```

```
[student@server -lab7 ~]$ gcc -o p3 pattern3_70.c
[student@server -lab7 ~]$ ./p3

enter the no. of rows: 5

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

Topic :

PIONEER
Page : 20
Date : 8 / 8 / 19

WAP to create an User Defined Function to convert a Decimal Number to its Binary Equivalent.

```
#include<stdio.h>
void decToBi(int n);
void main()
{
   int n;
   printf(" Enter a Decimal Number: ");
   scanf("%d", &n);
   printf("Binary Equivalent: ");
   decToBi(n);
}
void decToBi(int n)
{
    if(n==0)
       return;
    decToBi(n/2);
    printf("%d", (n%2));
}
```

```
[student@Server -lab7 ~] $ gcc -o con convert_70.c
[student@ Server - lab7 ~] $ ./con

enter a decimal Number: 7
Binary equivalent is : 111
```

WAP to print the following pattern where the number of lines is taken as input.

```
    *
   * *
  * * *
 * * * *
```

```
#include<stdio.h>
void main()
{
  printf (" Enter the no. of lines: ");
  int i, n;
  scanf("%d", &n);
  for (i=1; i<=n; i++)
  {
    for (j=i; j<=n; j++)
      printf (" ");
    for (j=1; j<=i; j++)
      printf (" * ");
    printf ("\n");
  }
}
```

```
[student@server -lab7 ~] $ gcc -o p2 pattern2-70.c
[student@server -lab7 ~] $ ./p2

enter the number of lines: 4

   *
  * *
 * * *
* * * *
```

WAP to take m-n numbers. Print and reverse the printed numbers.

```
#include <stdio.h>
void main()
{
  printf("Enter the Range as m-n :");
  int c, d, i, m, n, r;
  scanf("%d %d", &m, &n);
  printf("The Reversed Numbers are :\n");
  for(i=m; i<=n; i++)
  {
    c=i;
    r=0;
    while(c>0)
    {
      d= c%10;
      r = r*10+d;
      c= c/10;
    }
  printf("%d --> %d \n", i, r);
}
}
```

```
[student@server ~lab 7 ~]$gcc -o rev Reverse-70.c
[student@ server ~lab 7 ~] $ ./rev

enter the range m-n : 15  20
The reversed nos. are:

51  61  71  81  91  2
```

Topic :

PIONEER

Page : 23

Date : 8 / 8 / 19

WAP to print whether a number is positive or negative.

```
#include<stdio.h>
void main()
{
  printf ("Enter a Number:");
  int n;
  scanf ("%d", &n);
  if (n>=0)
    printf ("POSITIVE");
  else
    printf ("NEGATIVE");
}
```

29/08/19

```
[student@server ~lab7 ~]$ gcc -o pos positive-70.c
[student@server ~lab7 ~]$ ./pos

Enter a number: 5
Positive
```

Topic :

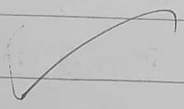
PIONEER

Page : 29

Date : 29/8/19

Write a Shell Program to take input of your name and then display it.

```
read n
echo "i am $n"
```



OUTPUT

```
[student@localhost Nishi]$ vim first.sh
[student@localhost Nishi]$ sh first.sh
[student@localhost Nishi]$ ./first.sh
Nishi
I am Nishi
[student@localhost Nishi]$
```

Topic :

Page : 25
Date : 28 / 9 / 19

Write a shell program to input your name and then display it to show the use of "double quotes (")", single quotes('), and no quotes.

```
read n
echo "i am $n"
echo 'i am $n'
echo i am $n.
```

OUTPUT

```
[student@localhost Nishi]$ vim second.sh
[student@localhost Nishi]$ sh second.sh
[student@localhost Nishi]$ ./second.sh
Nishi
i am Nishi
i am $n
i am Nishi
```

Write a shell Program to enter two numbers and display the sum, difference, quotient, product and remainder.

```
read a
read b
x = `expr $a + $b`
echo "sum is $x"
y = `expr $a - $b`
echo "Difference is $y"
z = `expr $a \* $b`
echo "Quotient is $z"
p = `expr $a \* $b`
echo "Product is $p"
q = `expr $a % $b`
echo "Remainder is $q"
```

OUTPUT

```
[student@localhost Nishi] $ vim third.sh
[student@localhost Nishi] $ sh third.sh
[student@localhost Nishi] $ ./third.sh
10
5
Sum is 15
Difference is 5
Quotient is 2
Product is 50
Remainder is 0
```

Topic :

PIONEER
Page : 27
Date : 28 / 9 / 19

Write a shell program to enter two numbers to compare them.

```
read a
read b
if [ $a -lt $b]
then
echo " $a is less than $b"
elif [$a -gt $b]
then
echo " $a is greater than $b"
else
echo " $a is equal to $b"
fi
```

OUTPUT

```
[student@localhost Nishi] $ vim fourth.sh
[student@localhost Nishi] $ sh fourth.sh
[student@localhost Nishi] $ ./fourth.sh
1
2
1 is less than 2
```

Write a shell program to print the first 10 natural numbers using for loop.

```
for ((i=1; i<=10; i++))
{
 echo "$i"
}
```

OUTPUT

```
[student@localhost Nishi] $ vim fifth.sh
[student@localhost Nishi] $ ch fifth.sh
[student@localhost Nishi] $ ./fifth.sh
1
2
3
4
5
6
7
8
9
10
```

Write a shell program to display the first 10 natural numbers using the while loop.

```
i=1
while [$i -le 10]
do
echo "$i"
i=`expr $i +1`
done
```

28/9/19

OUTPUT

```
[student@local host  Nishi] $ vim sixth.sh
[student @localhost  Nishi] $ sh sixth.sh
[student@ localhost  Nishi] $ ./sixth.sh
1
2
3
4
5
6
7
8
9
10
```

Topic :

PIONEER
Page : 30
Date : 19 / 9 / 19

Assignment-1

Write a shell script to print the following number pattern (using nested for loop).

```
        1
      2   2
    3   3   3
  4   4   4   4
5   5   5   5   5
```

```
echo " Enter the height of Pattern : "
read n
for ((i=1; i<=n ; i++))
{
K=' expr $n - $i'
for (( j=1; j<=K; j++))
{
echo -n" "
}
for (( j=1 ; j<=i ; j++))
{
echo -n " $i" .
}
echo
}
```

OUTPUT

```
[student@localhost ~]$ vim Assignment1.sh
[student@localhost ~]$ chmod 777 Assignment1.sh
[student@localhost ~]$./Assignment1.sh
Enter the Height of Pattern:
5
        1
      2   2
    3   3   3
  4   4   4   4
5   5   5   5   5
[student@localhost ~]$
```

Topic :

PIONEER
Page : 31
Date : 19 / 9 / 19

Assignment 2

Write a shell script to print the prime numbers between n and m. [n and m are user input] (using nested while loop).

```
echo "Enter the Range of n to m:"
read n
read m
i = $n
echo "The Prime Numbers in the Range:"
while [ $i -le $m]
do
c=0
j=1
while [ $j -le $i]
do
d = `expr $i % $j`
if [ $d -eq 0]
then
c=`expr $c +1`
fi
j=`expr $j +1`
done
if [$c- eq 2]
```

OUTPUT

```
[student@localhost ~] $ vim Assignment2.sh
[student@localhost ~] $ chmod 777 Assignment2.sh
[student@localhost ~] $ ./Assignment2.sh
Enter the Range of n to m:
1
5
The Prime Numbers in the Range:
2
3
5
```

```
then
echo "$i"
fi
i=`expr $i + 1`
done.
```

Assignment-3

Write a shell script to print all the fibonacci number between 1 and 100.

```
a=0
b=1
c=0
echo "The fibonacci Numbers between 1 and 100:"
while [ $c -le 100 ]
do
c=`expr $a + $b`
if [ $c -ge 100 ]
then
break
fi
echo "$c"
a=$b
b=$c
done
```

Topic :

PIONEER

Page : 34

Date : 19 / 9 / 19

Assignment 4

Write a shell script to check whether a user is logged in or not ?

```
echo "Enter the User Name :"
read n
x = `who | grep -wc "$n"
if [ $x -eq 0 ]
then
echo "User is Not Logged in"
else
echo "User Logged in"
fi
```

Topic :

PIONEER
Page : 35
Date : 26 / 9 / 19

Write a C program to show the use of thread by taking an integer as input and printing the value. Use sleep() function.

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<sys/types.h>
#include<pthread.h>
int i,f;
void *sum(void *);
int main()
{
  pthread_t tid;
  void *result;
  pthread_create(&tid, NULL, sum, (void *) NULL);
  sleep(2);
  printf("Enter the value of i:");
  scanf("%d", &i);
  f=1;
  pthread_join(tid, &result);
  printf("%s", (char *) result);
  return 0;
}
```

```
void *sum (void *a)
{
    while (f<1)
      sleep(2);
    printf ("%d\n", i);
    pthread_exit ("Done\n");
}
```

OUTPUT

```
[student@localhost ~] $ vim thread.c
[student@localhost ~] $ gcc thread.c -o t -lpthread -D_REENTRANT
[student@localhost ~] $./t
Enter the value of i: 5
5
Done
```

Topic :

PIONEER
Page : 37
Date : 26 / 9 / 19

Assignment 7

Write a C Program to create two thread and show that two thread are running quasi parallel with seperate thread id under single core CPU and check PID of the process and thread id of two threads.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <pthread.h>
int r=1;
void *sum(void *);
int main()
{
  pthread_t tid2;
  void *result;
  int i;
  pthread_create(&tid2, NULL, sum, (void *)NULL);
  for(i=0; i<5; i++)
  {
    if(r==1)
    {
      printf("%d %u %u \n", i, getpid(), pthread_self());
      r=0;
    }
```

```
        sleep(1);
    }
    pthread_join(tid2, &result);
    return 0;
}
void *sum(void *a1)
{
    int j;
    for(j=0; j<5; j++)
    {
        if(r==0)
        {
            printf("%d %u %u \n", j+2, getpid(), pthread_self());
            r=1;
        }
    sleep(1);
    }
    pthread_exit("HELLO\n");
}
```

OUTPUT

```
[student@localhost ~] $ vim Assignment7.c
[student@localhost ~] $ gcc Assignment7.c -o A7 -lpthread
                                          -D_REENTRANT

[student@localhost ~] $ ./A7
 0  5229    3079202496
 3  5229    3079199600
 1  5229    3079202496
 4  5229    3079199600
 2  5229    3079202496
 5  5229    3079199600
 4  5229    3079202496
 6  5229    3079199600
```

Topic :

PIONEER®
Page : 39
Date : 26 / 9 / 19

Assignment - 8

Write a C Program to create multiple thread in a single process and terminate ~~in~~ a that thread in reverse order.

```
#include<stdio.h>
#include <stdlib.h>
#include< string.h>
# include < sys/ types.h>
#include < pthread.h>
void * sum (int );
int main ()
{
  pthread_t tid2[10];
  int i;
  void *result2;
  printf ("creation \n");
  for (i=0; i<10; i++)
    pthread_create (&tid2[i], NULL, sum, (void *)i);
sleep(10);
printf (" Terminal");
for (i=9; i>=0; i--)
{
   pthread_join ( tid[i], &result2);
   printf ("%d\t %d \n", i, tid2[i]);
}
```

```
    return 0;
}
void *sum (int i)
{
    printf ("%d\t%d\t%d\n", i, getpid(), pthread_self());
    sleep(1);
    pthread_exit ("Done\n");
}
```

24.10.19

OUTPUT

```
[student@localhost ~] $ vim Assignment8.c
[student@localhost ~] $
[student@localhost ~] $ ./A8
creation
2   5416   -1232860304
1   5416   -1224467600
3   5416   -1241253008
0   5416   -1216074896
5   5416   -1258038416
4   5416   -1266431120
6   5416   -1291609232
Terminal
6   -1291609232
5   -1258038416
4   -1266431120
3   -1241253008
2   -1224467600
1   -1232860304
0   -1216074896
```

Assignment - 9

Write a c program using Thread concept to count the number of characters entered from the keyboard and also utilize the concept of Semaphore for thread synchronization.

```
#include <stdio.h>
#include <sys/types.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>
#include <semaphore.h>
void *count (void *arg);
sem_t s;
char area [1024];
int main()
{
  int ~~area~~ res;
  pthread_t tid 1;
  void *result;
  res = sum_init (&s, 0, 0);
  if (res != 0)
{
```

```
perror("semaphore creation failed");
exit (EXIT_FAILURE");
}
res = pthread_create (&Hid1, NULL, count, NULL);
if (res != 0)
{
   perror ("Thread creation Failed");
   exit ( EXIT_FAILURE);
}
printf (" Enter some Text. Enter 'end' to Finish \n");
while ( strncmp ("end", area, 3) != 0)
{
    fgets (area, 1024, stdin);
     sem_post (&s);
}
printf ("\n Waiting for Thread to Finish \n");
res = pthread_join (tid1, &result);
if (res != 0)
{
   perror ("Thread joined Failure");
   exit (EXIT_FAILURE);
}
printf (" Thread joined %s\n", (char *)result);
sem_destroy(&s);
exit (EXIT_SUCCESS);
```

```
}
void *count (void *arg)
{
    sem_wait (&s);
    while (strncmp ("end", area, 3)!=0)
    {
        printf (" You input %d character \n", strnlen (area)-1);
        sem_wait (&s);
    }
pthread_exit (" Thanks for CPU time and count
                function running with semaphore \n");
}
```

11/7/19

OUTPUT

```
[student@localhost ~] $ vim Assignment9.c
[student@localhost ~] gcc -o REENTRANT Assignment9.c -o A9
                                                -lpthread

[student@localhost ~] $ ./A9
Enter some text Enter 'end' to finish
Nishi
You input 5 characters
Helloo
You input 6 characters
end
Waiting for Thread to Finish
Thread joined
Thanks for CPU time and count function running with
semaphore
```

Topic :

PIONEER
Page : 44
Date : 8 / 11 / 19

Assignment - 10

Write a c program to invoke another program as new process and receive data from the invoked program.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
int main()
{
  FILE *rf;
  char buffer[1024];
  int res;
  memset(buffer, '\0', sizeof(buffer));
  rf = popen("ls -l","r");
  if(rf != NULL)
{
  res = fread(buffer, sizeof(char), 1024, rf);
  if(res>0)
{
printf("Output was for the prouss %d :- \n %s \n",
                                    getpid(), buffer);
}
pclose(rf);
```

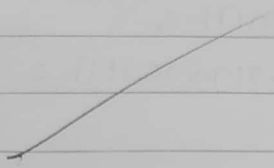
Topic :

PIONEER

Page : 45

Date : / /

```
    exit (EXIT_SUCCESS);
}
    exit (EXIT_FAILURE);
}
```



OUTPUT

```
[student@localhost ~] $ vim assignment10.c
[student@localhost ~] $ gcc -o a10 assignment10.c
[student@localhost ~] $ ./a10.
output was for the process 1373 :-
total 400
drwxrwxr-x. 2 student student 4096 Oct 25 11:49 25-10-11
-rwxrwxr-x. 1 student 5400 Nov 7 15.03 a10
-rwxrwxr-x. 1 student 6361 Nov 7 14:58 a11
-rwxrwxr-x 1 student 6592 Nov 5 12:31 ab
-rw-rw-r-. 1 student 158 Aug 20 14:43 assignment3.sh
[student@localhost ~] $.
```

Topic :

PIONEER®
Page : 46
Date : 8 / 11 / 19

Assignment - 11

Write a C Program which will create child prouss using fork() and Parent prouss and chid prouss data through unnamed pipe.

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
int main ()
{
 int res;
 int pipes[2];
 char msg[1024];
 char buffer[ BUFSIZ+1];
 pid-t pid;
 printf ("Enter Data\n");
 scanf ("%s", &msg);
 memset (buffer, '\0', sizeof (buffer));
  if (pipe(pipes) ==0)
{
  pid = fork();
  if (pid == -1)
{
```

```
        fprintf ( stderr , " Took failure ");
        exit (EXIT_FAILURE);
    }
    if (pid == 0)
    {
    res = read (pipes [0], buffer, BUFSIZ);
    printf (" prouss id %d Read %d bytes : %s from parent
            prouss %d \n", getpid() , rus, buffer, getppid()));
    exit (EXIT_SUCCESS);
    }
    else
    {
    res = write (pipes [1], msg, strlen (msg));
    printf (" Prouss id %d wrote %d bytes \n", getpid() , rus);
    sleep (10);
    }
}
    exit (EXIT_SUCCESS);
}
```

OUTPUT

```
[student@localhost ~] $ vim assignment11.c
[student@localhost ~] $ gcc -o a11 assignment11.c
[ student@localhost ~] $ ./a11
Enter Data
nishi
Proussid 1232 wrote 5 bytes
Procssid 1233 Read 5 bytes: nishi from parent prouss
                                    1232
```

Topic :

PIONEER

Page : 48

Date : 8 / 11 / 19

Assignment - 12

Write the reader's and Writer's programme using named pipe to communicate with each other.

FOR WRITING DATA (PRODUCER)

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <fcntl.h>
#include <sys/types.h>
#include <sys/stat.h>
int main()
{
int pipe_fd;
int res;
int s=0;
char buffer[1024];
printf("Enter data\n");
fgets(buffer, 1024, stdin);
res = mkfifo("fifo3", 0777);
printf("Process %d opening FIFO O_WRONGLY\n", getpid
pipe_fd = open("fifo3", O_WRONLY);
```

```
printf (" Prouss %o result %d \n", getpid(), pide_fd);
ores = write (pipe_fd, buffer, sizeof(buffer));
(void) close (pipe_fd);
unlink ("fifo3");
printf (" Prouss %d finished \n", getpid());
exit ( EXIT_SUCCESS);
}
```

FOR READING DATA (CONSUMER)

```
#include<unistd.h>
#include <stdlib.h>
#include<stdio.h>
# include < string.h>
# include < fcntl.h>
# include < sys/ types.h>
# include <sys/ stat.h>
int main()
{
int fd;
int r, l;
char buffer [1024];
printf (" Prouss %d opening FIFO O_RDONLY \n", getpid()
fd = open ("fifo3", O_RDONLY);
printf (" Prouss %o result %d \n", getpid(), fd);
```

Topic :

PIONEER

Page : 50

Date : / /

```
r= read(fd, buffer, sizeof(buffer));
l= strlen(buffer)-1;
(void) close(fd);
unlink("fifo3");
printf("Process %ld finished, with reading %ds of
        %ld bytes\n", getpid(), buffer, l);
exit(EXIT_SUCCESS);
}
```

14/11/19.

OUTPUT

```
[student@localhost ~]$ vim assignment12a.c
[student@localhost ~]$ gcc -o a12a assignment12a.c
[student@localhost ~]$ ./a12a
 Enter data
 nishi
Process  1578  opening FIFO. O_WRONLY
Process  1578  result 3
Process  1578  finished
```

OUTPUT

```
[student@localhost ~] $ vim assignment12b.c
[student@localhost ~] $ gcc -o assign12b assignment12b.c
[student@localhost ~] $./assign12b
Process 1640 opening FIFO O_RDONLY
Process 1640 finished, with reading nishi of 5 byts
```