Name : Nisarg .k. Amlani
Roll : Ce001
Id : 22ceueg082

Lab : 10

---

1. Write header.h file and include process_fork() and process_join() and shared() functions.

```
#include<unistd.h>
#include<stdlib.h>
#include<sys/wait.h>
#include<sys/shm.h>
#include<sys/ipc.h>
int process_fork (int nproc)
{
   int j;
   for (j=1;j<nproc; j++)
   {
       if (fork()==0)
       return (j);
   }
   return(0);
}

void process_join (int nproc, int id)
{
   int i;
   if (id==0)
   {
       for (i=1;i<nproc; i++)
           wait(0);
   }
   else
       exit(0);
}
```

```
char *shared (int size, int *shmid)
{
   *shmid =shmget(IPC_PRIVATE,size,0666|IPC_CREAT);
   return (shmat(*shmid,0,0));
}
```

2. WAP to create n number of processes.

```
#include <stdio.h>
#include "header.h"

int main()
{
   int id, nproc;

   printf("Enter number of processes you want to create: ");
   scanf("%d", &nproc);

   id = process_fork(nproc);

   printf("Process id is %d\n", id);

   process_join(nproc, id);

   printf("Parent id is %d\n", id);

   return 0;
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
Enter number of processes you want to create: 5
Process id is 1
Process id is 2
Process id is 3
Process id is 0
Process id is 4
Parent id is 0

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
```

3. Write a parallel program to add four variables (a,b,c,d)

```
#include "header.h"
#include<stdio.h>
#include<stdlib.h>
int main(){
  int id, nproc;
  int sum1 = 0, *sum2 , sum=0, shmid;
  sum2 = (int *)shared(sizeof(int),&shmid);
  *sum2 = 0;
  int a=1, b=10, c=9, d=20;
  id = process_fork(2);

  if(id==0){
      sum1 = a+b;
      printf("sum by Parent process : %d\n", sum1);
  }
  else{
      *sum2 = c+d;
      printf("sum by process %d  : %d\n", id, *sum2);
  }
  process_join(2,id);
  sum = sum1 + *sum2;
```

```
  printf("Final sum %d\n", sum);
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
sum by Parent process : 11
sum by process 1  : 29
Final sum 40

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
```

4. Write a parallel program to copy one array into another using loop splitting.

```
#include "header.h"
#include <stdio.h>
int main()
{
 int nproc = 4;
 int id, a[10], *b, i;
 b = (int *)shared(sizeof(i) * 10, &i);
 for (i = 0; i < 10; i++)
 {
     printf ("Enter elements of a [%d]:",i+1);
     scanf ("%d", &a [i]);
 }
 id = process_fork(nproc);
 if (id == 0)
 {
     for (i = 0; i < 10; i += nproc){
         printf("Element %d is copied by process %d\n", i, id);
         *(b + i) = a[i];
     }
```

```
}
else if (id == 1)
{
    for (i = 1; i <= 10; i += nproc){
        printf("Element %d is copied by process %d\n", i, id);
        *(b + i) = a[i];
    }
}
else if (id == 2)
{
    for (i = 2; i < 10; i += nproc){
        printf("Element %d is copied by process %d\n", i, id);
        *(b + i) = a[i];
    }
}
else
{
    for (i = 3; i <= 10; i += nproc){
        printf("Element %d is copied by process %d\n", i, id);
        *(b + i) = a[i];
    }
}


process_join(nproc, id);
printf("elements of b :");
for(int i=0;i<10;i++){
    printf("%d, ", b[i]);
}
printf("\n");
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
Enter elements of a [1]:1
Enter elements of a [2]:2
Enter elements of a [3]:3
Enter elements of a [4]:4
Enter elements of a [5]:5
Enter elements of a [6]:6
Enter elements of a [7]:7
Enter elements of a [8]:8
Enter elements of a [9]:9
Enter elements of a [10]:1
Element 1 is copied by process 1
Element 5 is copied by process 1
Element 9 is copied by process 1
Element 2 is copied by process 2
Element 6 is copied by process 2
Element 0 is copied by process 0
Element 4 is copied by process 0
Element 8 is copied by process 0
Element 3 is copied by process 3
Element 7 is copied by process 3
elements of b :1, 2, 3, 4, 5, 6, 7, 8, 9, 1,

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
```

5. Write a parallel program to copy one array into another using loop splitting
and strictly using one loop.

```c
#include "header.h"
#include <stdio.h>
int main()
{
   int nproc = 4;
   int id, a[10], *b, i;
   b = (int *)shared(sizeof(i) * 10, &i);
   for (i = 0; i < 10; i++)
   {
       printf("Enter elements of a [%d]:", i + 1);
       scanf("%d", &a[i]);
```

```
    }
    id = process_fork(nproc);
    for (int i = id; i < 10; i += nproc)
    {
        printf("Element %d is copied by process %d\n", i, id);
        *(b + i) = a[i];
    }

    process_join(nproc, id);
    printf("elements of b :");
    for (int i = 0; i < 10; i++)
    {
        printf("%d, ", b[i]);
    }
    printf("\n");
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
Enter elements of a [1]:1
Enter elements of a [2]:2
Enter elements of a [3]:3
Enter elements of a [4]:4
Enter elements of a [5]:5
Enter elements of a [6]:6
Enter elements of a [7]:7
Enter elements of a [8]:8
Enter elements of a [9]:9
Enter elements of a [10]:1
Element 1 is copied by process 1
Element 5 is copied by process 1
Element 9 is copied by process 1
Element 2 is copied by process 2
Element 6 is copied by process 2
Element 0 is copied by process 0
Element 4 is copied by process 0
Element 8 is copied by process 0
Element 3 is copied by process 3
Element 7 is copied by process 3
elements of b :1, 2, 3, 4, 5, 6, 7, 8, 9, 1,

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
```

6. Write a parallel program to do matrix addition with loop splitting.

```c
#include "header.h"
#include <stdio.h>
int main()
{
  // Declaration and scan values
  int id, a[3][3], b[3][3], *c, i, j, nproc = 3, shmid;
  c = (int *)shared(sizeof(int) * 3 * 3, &shmid);
  // c is shared among parent and children
  for (j = 0; j < 3; j++) // Scan matrix A
  {
      for (i = 0; i < 3; i++)
```

```
      {
          printf ("Enter element a [%d] [%d]:",j+1,i+1);
          scanf ("%d", &a [j] [i]);
      }
  }
  printf("\n");

  for (j = 0; j < 3; j++) // Scan matrix B
  {
      for (i = 0; i < 3; i++)
      {
          printf("Enter element b [%d] [%d]= ", j + 1, i + 1);
          scanf("%d", &b[j][i]);
      }
  }

  // program logic
  id = process_fork(nproc); // nproc=3
  for (i = id; i < 3; i += 3)
  {
      for (j = 0; j < 3; j++){
          *(c + 3 * i + j) = a[i][j] + b[i][j];
          printf("Element %d is added using process id: %d\n", *(c+3*i+j),
id);
      }
  }

  process_join(nproc, id); // Children are exited
   // Print final Values
  for (i = 0; i < 3; i++) // print matrix A
  {
      for (j = 0; j < 3; j++)
      {
          printf("a [%d][%d]=%d \t",i+1,j+1, a[i][j]);
      }
      printf("\n");
  }
  printf("\n");

  for (i = 0; i < 3; i++) // print matrix B
```

```
  {
      for (j = 0; j < 3; j++)
      {
          printf("b [%d][%d]=%d \t",i+1,j+1, b[i][j]);
      }
      printf("\n");
  }
  printf("\n");
   for (i = 0; i < 3; i++) // print resultant matrix C
  {
      for (j = 0; j < 3; j++)
          printf("c [%d][%d]=%d\t", i + 1, j+1, *(c + 3 * i + j));
      printf("\n");
  }
}
```

```
a [1][1]=1      a [1][2]=2      a [1][3]=3
a [2][1]=4      a [2][2]=5      a [2][3]=6
a [3][1]=7      a [3][2]=8      a [3][3]=9

b [1][1]=1      b [1][2]=2      b [1][3]=3
b [2][1]=4      b [2][2]=5      b [2][3]=6
b [3][1]=7      b [3][2]=8      b [3][3]=9

c [1][1]=2      c [1][2]=4      c [1][3]=6
c [2][1]=8      c [2][2]=10     c [2][3]=12
c [3][1]=14     c [3][2]=16     c [3][3]=18

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$
```

7. Write a parallel program to do matrix multiplication with loop splitting.

```
#include "header.h"
#include <stdio.h>
```

```
int main()
{
   int m = 3;
   int mat1[][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
   int mat2[][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
   int id, nproc = 3, shmid, i;
   int *mat3;

   // Allocate shared memory for the result matrix (mat3)
   mat3 = (int *)shared(sizeof(int) * m * m, &shmid);

   // Print mat1
   printf("Matrix 1:\n");
   for(int i = 0; i < m; i++) {
       for(int j = 0; j < m; j++) {
           printf("%d ", mat1[i][j]);
       }
       printf("\n");
   }

   // Print mat2
   printf("Matrix 2:\n");
   for(int i = 0; i < m; i++) {
       for(int j = 0; j < m; j++) {
           printf("%d ", mat2[i][j]);
       }
       printf("\n");
   }

   // Fork processes for matrix multiplication
   id = process_fork(nproc);

   // Perform matrix multiplication in parallel
   for(i = id; i < m; i += nproc) {
       for(int j = 0; j < m; j++) {
           int sum = 0;
           for(int k = 0; k < m; k++) {
               sum += mat1[i][k] * mat2[k][j];
           }
           *(mat3 + (m * i) + j) = sum;
```

```
            printf("Process %d : %d\n", id, *(mat3 + (m * i) + j));
        }
    }

    // Join processes after multiplication
    process_join(nproc, id);

    // Print final result matrix mat3
    printf("\nFinal Matrix 3 (Result):\n");
    for(int i = 0; i < m; i++) {
        for(int j = 0; j < m; j++) {
            printf("%d ", *(mat3 + (m * i) + j));
        }
        printf("\n");
    }

    return 0;
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
Matrix 1:
1 2 3
4 5 6
7 8 9
Matrix 2:
1 2 3
4 5 6
7 8 9
Process 1 : 66
Process 1 : 81
Process 1 : 96
Process 0 : 30
Process 0 : 36
Process 0 : 42
Process 2 : 102
Process 2 : 126
Process 2 : 150

Final Matrix 3 (Result):
30 36 42
66 81 96
102 126 150

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$
```

8. Write a parallel program to find factorial of a number using loop splitting.

```c
#include <stdio.h>
#include "header.h"
int main(){
 int num;
 printf("Enter a number: ");
 scanf("%d", &num);
 int id, nproc = 4, shmid, i;
 int *fac, final_factorial = 1;
 fac = (int *)shared(sizeof(int)*nproc, &shmid);
 id = process_fork(nproc);
 for(i = id; i<nproc; i+=nproc)
```

```
{
    *(fac + i) = 1;
}
for(i = id +1; i<=num; i += nproc)
{
    *(fac + id) *= i;
}
printf("Process %d : %d\n", id, *(fac + id));
process_join(nproc, id);
for(i = 0; i<nproc; i++)
{
    final_factorial *= *(fac + i);
}
 printf("Factorial of %d is %d\n", num, final_factorial);
}
```

```
┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$ ./a.out
Enter a number: 5
Process 1 : 2
Process 2 : 3
Process 0 : 5
Process 3 : 4
Factorial of 5 is 120

┌──(nisarg㉿fedora)-[~/…/Sem_6_repo/ACA/ACA_10/ACA_10]
└─$
```