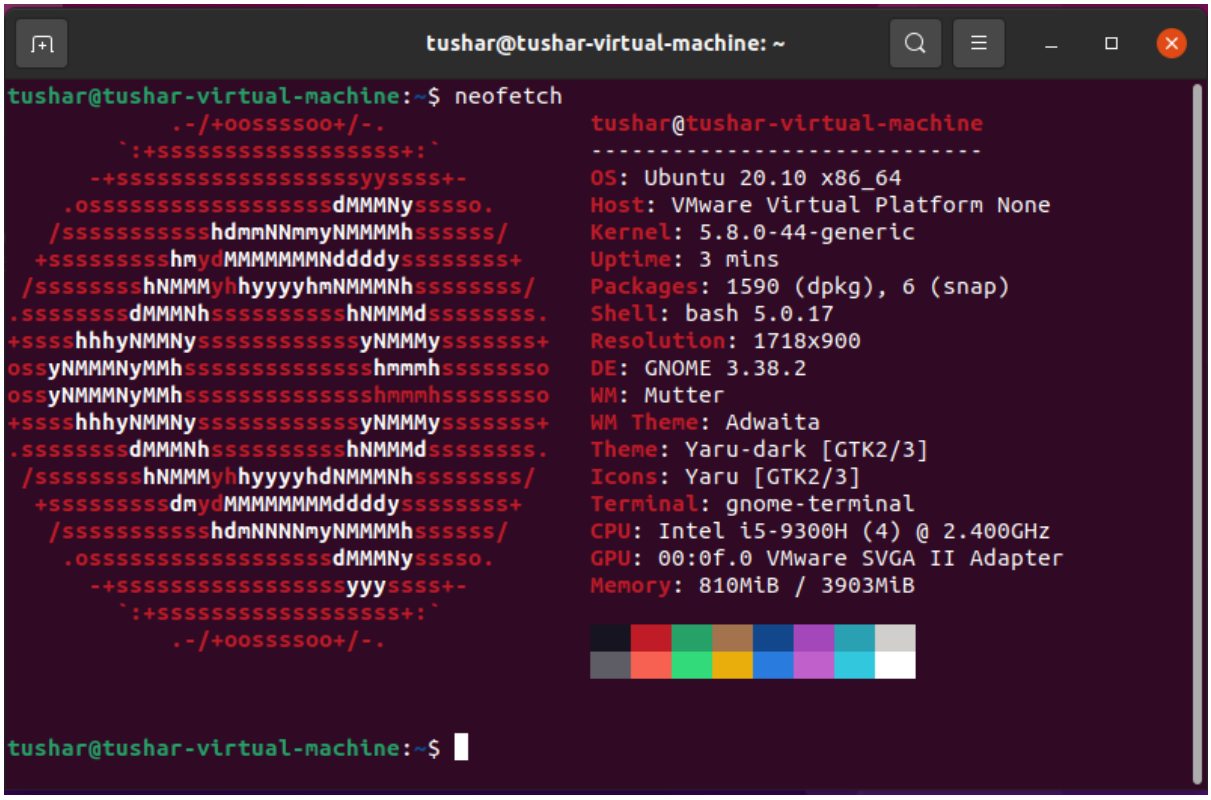
OS Experiments

RA1911003010702

Tushar Swarup Tandon

1→ Linux Installation



2→Linux File Commands

Q1. Write a command to cut 5 to 8 characters of the file f1.
$cut -c 5-8 f1

Q2. Write a command to display user-id of all the users in your system.
$cat /etc/passwd

Q3. Write a command to paste all the lines of the file f1 into single line
$paste -s f1

Q4. Write a command to cut the first field of file f1 and second field of file f2 and paste into the file f3.
$paste <(cut -f 1 f1) <(cut -f 2 f2) >f3

Q5. Write a command to change all small case letters to capitals of file f2.
$tr [‘a’-’z’][‘A’-’Z’] < f2

Q6. Write a command to replace all tab character in the file f2 by :
$tr ‘\t’’:’ <f2

Q7. Write a command to check whether the user judith is available in your system or not. (use grep)
$grep -c ‘^judith’ /etc/passwd

Q8. Write a command to display the lines of the file f1 starts with SRM.
$grep “^SRM” f2

Q9. Write a command to display the name of the files in the directory /etc/init.d that contains the pattern grep.
$grep “grep” /etc/init.d/

Q10. Write a command to display the names of nologin users. (Hint: the command nologin is specified in the last filed of the file /etc/passwd for nologin users)
$grep “nologin$” /etc/passwd

Q11. Write a command to sort the file /etc/passwd in descending order
$file /etc/passwd | sort -r:

Q12. Write a command to sort the file /etc/passwd by user-id numerically. (Hint : user-id is in 3rd field)
$file /etc/passwd | sort -n -t: +2

Q13. Write a command to sort the file f2 and write the output into the file f22. Also eliminate duplicate lines.
$sort -u -o f22 f2

Q14. Write a command to display the unique lines of the sorted file f21. Also display the number of occurrences of each line.
$uniq f21
$uniq -c f21

Q15. Write a command to display the lines that are common to the files f1 and f2.
$comm f1 f2

3→ C program and Process Creation

Q1.

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi cQuestion1.c
tushar@tushar-virtual-machine:~$ gcc cQuestion1.c
tushar@tushar-virtual-machine:~$ ./a.out
Before fork a=5 b=10
In child a=6 b=11
In Parent a=4 b=9
tushar@tushar-virtual-machine:~$
```

Q2.

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi cQuestion1.c
tushar@tushar-virtual-machine:~$ gcc cQuestion1.c
tushar@tushar-virtual-machine:~$ ./a.out
Before fork a=5 b=10
In child a=6 b=11
In Parent a=-1 b=-817992705
Segmentation fault (core dumped)
tushar@tushar-virtual-machine:~$
```

Q3.

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi j.c
tushar@tushar-virtual-machine:~$ gcc j.c
tushar@tushar-virtual-machine:~$ ./a.out
SRMIST
SRMIST
SRMIST
tushar@tushar-virtual-machine:~$ SRMIST
SRMIST
SRMIST
SRMIST
SRMIST
```

Q4.

tushar@tushar-virtual-machine: ~

```c
#include <stdio.h>
#include<unistd.h>
int main()
{
    int pid,n,oddsum=0,evensum=0;
    printf("Enter the value of n : ");
    scanf("%d",&n);
    pid=fork();
    if(pid > 0)
    {
        for(int i = 1; i <= n; i = i + 2)
        {
            oddsum += i;
         }
         printf("Sum of odd numbers  : %d\n", oddsum);
    }
    else
    {
        for(int i = 0; i <= n; i= i + 2)
        {
            evensum += i;
        }
        printf("Sum of even numbers : %d\n", evensum);
    }
    return 0;
}
~
```

```
tushar@tushar-virtual-machine:~$ vi q4.c
tushar@tushar-virtual-machine:~$ gcc q4.c
tushar@tushar-virtual-machine:~$ ./a.out
Enter the value of n : 10
Sum of odd numbers   : 25
Sum of even numbers  : 30
tushar@tushar-virtual-machine:~$
```

Q5. 2^n - 1 child processes will be created

Q6.

```
#include <stdio.h>
#include <unistd.h>
int main()
{

        int i;
        i=fork();
        if(i==0)
        {

                printf("In Child Process\n");
                printf("Parent Process ID : %d\nChild Process ID : %d\n",getppid(),getpid());
        }
        else
        {
                printf("In Parent Process\n");
                printf("Parent Process ID : %d\nChild Process ID : %d\n",getppid(),getpid());
                printf("\n");

        }

        return 0;

}
tushar@tushar-virtual-machine:~$ vi q6.c
tushar@tushar-virtual-machine:~$ gcc q6.c
tushar@tushar-virtual-machine:~$ ./a.out
In Parent Process
Parent Process ID : 2882
Child Process ID : 3058

In Child Process
Parent Process ID : 3058
Child Process ID : 3059
tushar@tushar-virtual-machine:~$
```

Q7.

```
#include <stdio.h>
#include<unistd.h>
int main()
{
    fork();
    fork()&&fork()||fork();
    fork();
    if(!fork())
        printf("Yes ");
    return 0;
}
tushar@tushar-virtual-machine:~$ vi t.c
tushar@tushar-virtual-machine:~$ gcc t.c
tushar@tushar-virtual-machine:~$ ./a.out
Yes tushar@tushar-virtual-machine:~$ Yes Yes Yes Yes Yes Yes Yes Yes Yes Yes Ye
s Yes Yes Yes Yes Yes Yes Yes Yes
```

19 child processes will be created.

4→System Admin Commands

```
tushar@tushar-virtual-machine:~$ sudo adduser t
Adding user `t' ...
Adding new group `t' (1001) ...
Adding new user `t' (1001) with group `t' ...
The home directory `/home/t' already exists.  Not copying from `/etc/skel'.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for t
Enter the new value, or press ENTER for the default
        Full Name []: tushar
        Room Number []: 702
        Work Phone []: 0
        Home Phone []: 0
        Other []: 0
Is the information correct? [Y/n] y
tushar@tushar-virtual-machine:~$ sudo users
tushar
tushar@tushar-virtual-machine:~$ sudo deluser t
Removing user `t' ...
Warning: group `t' has no more members.
Done.
tushar@tushar-virtual-machine:~$
```

5→Simple Task Automation

```
tushar@tushar-virtual-machine:~$ sudo service cron start
tushar@tushar-virtual-machine:~$ date
Sunday 02 May 2021 03:52:16 AM IST
tushar@tushar-virtual-machine:~$ tty
/dev/pts/1
tushar@tushar-virtual-machine:~$ crontab -e
crontab: installing new crontab
tushar@tushar-virtual-machine:~$ SRM
```

```
tushar@tushar-virtual-machine: ~
  GNU nano 5.2                    /tmp/crontab.PT3J27/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
57 03 2 5 * echo "SRM" >/dev/pts/1

                                [ Read 24 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^_ Go To Line
```

6→ Simple bash Scripts

i)Printing hello world

```
tushar@tushar-virtual-machine: ~/OS Lab
  GNU nano 5.2                    week6Act1.c                    Modified
#!/bin/bash
echo "Hello World"

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

```
tushar@tushar-virtual-machine:~/OS Lab$ nano week6Act1.c
tushar@tushar-virtual-machine:~/OS Lab$ bash week6Act1.c
Hello World
tushar@tushar-virtual-machine:~/OS Lab$
```

ii)Simple Operations

```
tushar@tushar-virtual-machine: ~/OS Lab
  GNU nano 5.2                    week6Act2.sh                   Modified
#!/bin/bash
((sum=25+25))
echo $sum

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

```
tushar@tushar-virtual-machine:~/OS Lab$ nano week6Act2.sh
tushar@tushar-virtual-machine:~/OS Lab$ bash week6Act2.sh
50
tushar@tushar-virtual-machine:~/OS Lab$
```

3→use of While Loop

```
tushar@tushar-virtual-machine: ~/OS Lab
  GNU nano 5.2                    week6Act3.sh                    Modified
#!/bin/bash
valid=true
count=1
while [ $valid ]
do
     echo $count
     if [ $count -eq 5 ];
     then
          break
     fi
          ((count++))
done

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

```
tushar@tushar-virtual-machine:~/OS Lab$ nano week6Act3.sh
tushar@tushar-virtual-machine:~/OS Lab$ bash week6Act3.sh
1
2
3
4
5
tushar@tushar-virtual-machine:~/OS Lab$
```

7→Shell programming

1) a) Present working Directory has a list of files and directories. Write a shell script to list only the name of sub directories in the present working Directory – Simple if .. fi statement.

Code:

```
tushar@tushar-virtual-machine: ~
#!/bin/sh
for i in *
do
    if [[ -d $i ]]
    then
          echo $i
    fi
done
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"week7Act1.sh" 8 lines, 82 characters
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act1.sh
tushar@tushar-virtual-machine:~$ bash week7Act1.sh
Desktop
Documents
Downloads
Music
OS Lab
Pictures
Public
Templates
Videos
tushar@tushar-virtual-machine:~$
```

1) b) Every directory and file created have a permission for directory, user , group and others. Every user, group and others have all the three permissions as read, write and execute. Write a bash shell script to read a filename from the shell and check whether the file has execute permission or not. If not, add the permission – if..else..fi statement.

Code:

```
tushar@tushar-virtual-machine: ~
#!/bin/sh
echo "Enter Filename"
read fname
if [ -x $fname ]
then
     echo "Yes $fname has permission to execute"
else
     chmod +x $fname
fi
~
~
~
~
~
~
~
~
~
~
~
~
~
"week7Act2.sh" 9 lines, 144 characters
```

Output :

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act2.sh
tushar@tushar-virtual-machine:~$ bash week7Act2.sh
Enter Filename
Downloads
Yes Downloads has permission to execute
tushar@tushar-virtual-machine:~$ bash week7Act2.sh
Enter Filename
week7Act1.sh
tushar@tushar-virtual-machine:~$ bash week7Act2.sh
Enter Filename
week7Act1.sh
Yes week7Act1.sh has permission to execute
tushar@tushar-virtual-machine:~$
```

1) c) Write a shell script to print a greeting as specified below.

If hour is greater than or equal to 0 (midnight) and less than or equal to 11 (up to 11:59:59), "Good morning" is displayed.
If hour is greater than or equal to 12 (noon) and less than or equal to 17 (up to 5:59:59 p.m.), "Good afternoon" is displayed.
If neither of the preceding two conditions is satisfied, "Good evening" is displayed.

Code:

```
tushar@tushar-virtual-machine: ~
h=`date +%H`

if [ $h -lt 12 ]; then
  echo Good morning
elif [ $h -lt 18 ]; then
  echo Good afternoon
else
  echo Good evening
fi

~
~
~
~
~
~
~
~
~
~
~
~
"week7Act3.sh" 10 lines, 133 characters
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act3.sh
tushar@tushar-virtual-machine:~$ date
Tuesday 23 March 2021 10:34:28 PM IST
tushar@tushar-virtual-machine:~$ bash week7Act3.sh
Good evening
tushar@tushar-virtual-machine:~$
```

1) d) Write a shell script to simulate a online personal assistant for banking as specified below.
Use the following input keywords to get online assistance from bank
1) If keyword is loan then ask for “Type of Loan”
2) If keyword is personalloan or personal_loan or pl then print reply as “Currently no offer for personal loan”
3) If keyword is carloan or car_loan then print reply as “Currently best offer of 7 % is going on!”
4) If keyword is home or housingloan or housing_loan or homeloan or home_loan then print reply as “Currently best offer of 8 % is going on!”
5) If keyword is card or credit or creditcard or debit or debitcard then print reply as “Contact nearest branch!”
6) If keyword is bye print reply as “See you later!”
7) If keyword is other than above keywords then print reply as “Sorry, I don't understand!”

Code:

```
tushar@tushar-virtual-machine: ~
echo "May i help you"
read str
echo "Type of loan?"
if [[ "$str" == "loan" ]]
then
  read str1
  until [[ "$str1" == "bye" ]]
    do
      case "$str1" in
       "personal") echo "Currently no offer for personal loan";;
        "personalloan") echo "Currently no offer for personal loan";;
        "personal_loan") echo "Currently no offer for personal loan";;
        "car") echo "Currently best offer of 7 % is going on!";;
        "carloan") echo "Currently best offer of 7 % is going on!";;

        "car_loan") echo "Currently best offer of 7 % is going on!";;
        "home") echo "Currently best offer of 8 % is going on!";;
        "housingloan") echo "Currently best offer of 8 % is going on!";;
        "housing_loan") echo "Currently best offer of 8 % is going on!";;
        "homeloan") echo "Currently best offer of 8 % is going on!";;
        "home_loan") echo "Currently best offer of 8 % is going on!";;
        "card") echo "Contact nearest branch!";;
        "credit") echo "Contact nearest branch!";;
        "credit_card") echo "Contact nearest branch!";;
        "debit") echo "Contact nearest branch!";;
        "debit_card") echo "Contact nearest branch!";;
        "bye") echo "see you again";;
         *) echo "Sorry, I don't understand";;

       esac
      read str1
    done
echo "see you again"
  echo "that's all folks!"
else
  echo "that's all folks!"
fi
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act4.sh
tushar@tushar-virtual-machine:~$ bash week7Act4.sh
May i help you
loan
Type of loan?
car
Currently best offer of 7 % is going on!
home
Currently best offer of 8 % is going on!
personal
Currently no offer for personal loan
education
Sorry, I don't understand
card
Contact nearest branch!
credit
Contact nearest branch!
debit
Contact nearest branch!
bye
see you again
that's all folks!
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
```

2) a) Bash Shell Script to find whether the given number is Armstrong number - While Loop

Code:

```
tushar@tushar-virtual-machine: ~
while [ $n -gt $b ]
do
r=$((n % c))
i=$((r * r * r))
s=$((s + i))
n=$((n / c))
done
if [ $s == $t ]
then
echo "It is an Amstrong number"
else
echo "It is not an Armstrong number"
fi
}

result=`ams $n`
echo "$result"
~
~
~
~
~
~
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act5.sh
tushar@tushar-virtual-machine:~$ bash week7Act5.sh
Enter the number
6
It is not an Armstrong number
tushar@tushar-virtual-machine:~$ bash week7Act5.sh
Enter the number
153
It is an Amstrong number
tushar@tushar-virtual-machine:~$ bash week7Act5.sh
Enter the number
154
It is not an Armstrong number
tushar@tushar-virtual-machine:~$
```

2) b) Bash Shell Script to display the weekdays and weekends - FOR loop

Code:

```
tushar@tushar-virtual-machine: ~
i=1
for day in Mon Tue Wed Thu Fri Sat Sun
do
 if(( i>5 ))
 then
  echo "Day $((i++)) : $day (WEEKEND)"
else
   echo "Day $((i++)) : $day (weekday)"
 fi
done
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"week7Act6.sh" 10 lines, 158 characters
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act6.sh
tushar@tushar-virtual-machine:~$ bash week7Act6.sh
Day 1 : Mon (weekday)
Day 2 : Tue (weekday)
Day 3 : Wed (weekday)
Day 4 : Thu (weekday)
Day 5 : Fri (weekday)
Day 6 : Sat (WEEKEND)
Day 7 : Sun (WEEKEND)
tushar@tushar-virtual-machine:~$
```

2) c) Bash Shell Script to display the perfect numbers from a given range of numbers - UNTIL loop

Code:

```
tushar@tushar-virtual-machine: ~
echo Enter a starting value
read x
echo Enter a Ending value
read y
for(( no=x;no<=y;no++ ))
do
i=1
ans=0
while [ $i -le `expr $no / 2` ]
do
  if [ `expr $no % $i` -eq 0 ]
  then
    ans=`expr $ans + $i`
   fi
  i=`expr $i + 1`
done
if [ $no -eq $ans ]
then
  echo $no is a perfect number
fi
done
~
~
~
"week7Act7.sh" 21 lines, 298 characters
```

Output:

```
tushar@tushar-virtual-machine: ~
tushar@tushar-virtual-machine:~$ vi week7Act7.sh
tushar@tushar-virtual-machine:~$ bash week7Act7.sh
Enter a starting value
1
Enter a Ending value
100
6 is a perfect number
28 is a perfect number
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
tushar@tushar-virtual-machine:~$
```

8→Process Creation

1. Implement the C program in which the child process calculates the sum of odd numbers and the parent process calculate the sum of even numbers up to the number 'n'.

Code:

tushar@tushar-virtual-machine: ~/OS Lab/week8

```c
#include <unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int k;
    printf("Enter the value of n: ");
    scanf("%d", &k);
    int odd = 0, even = 0, n, i;
    n = fork();
    if(n > 0)
    {
        for(i = 0; i <= k; i++)
        {
            if(i % 2 == 0)
            {
                even += i;
            }
        }
        printf("Even sum = %d\n", even);
    }
    else
    {
        for(i = 0; i <= k ; i++)
        {
            if(i % 2 == 1)
            {
                odd += i;
            }
        }
        printf("Odd sum = %d\n", odd);
    }
    return 0;
}
```

Output:

tushar@tushar-virtual-machine: ~/OS Lab/week8

```
tushar@tushar-virtual-machine:~/OS Lab/week8$ vi week8Act1.c
tushar@tushar-virtual-machine:~/OS Lab/week8$ gcc week8Act1.c
tushar@tushar-virtual-machine:~/OS Lab/week8$ ./a.out
Enter the value of n: 100
Odd sum = 2500
Even sum = 2550
tushar@tushar-virtual-machine:~/OS Lab/week8$
```

2. Implement the C program in which main program accepts the integers to be sorted Main program uses the fork system call to create a new process called a child process. Parent process sorts the integers using insertion sort and waits for child process using wait system call to sort the integers using selection sort.

Code:

tushar@tushar-virtual-machine: ~/OS Lab/week8

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>

int split (int[], int, int);
void selectionSort (int *, int, int);

void insertionSort (int arr[], int low, int mid, int high)
{
    int i, j, k, l, b[20];
    l = low;
    i = low;
    j = mid + 1;
     while ((l <= mid) && (j <= high))
    {
        if (arr[l] <= arr[j])
            {
            b[i] = arr[l];
            l++;
        }
        else
            {
            b[i] = arr[j];
            j++;
        }
        i++;
    }
    if (l > mid)
    {
        for (k = j; k <= high; k++)
            {
            b[i] = arr[k];
            i++;
        }
    }
    else
    {
        for (k = l; k <= mid; k++)
            {
            b[i] = arr[k];
            i++;
        }
    }
```

tushar@tushar-virtual-machine: ~/OS Lab/week8

```
    for (k = low; k <= high; k++)
    {
        arr[k] = b[k];
    }
}

void partition (int arr[], int low, int high)
{
    int mid;
    if (low < high)
    {
        double temp;
        mid = (low + high) / 2;
        partition (arr, low, mid);
        partition (arr, mid + 1, high);
        insertionSort (arr, low, mid, high);
    }
}

void display (int a[], int size)
{
    int i;
    for (i = 0; i < size; i++)
    {
        printf ("%d\t\t", a[i]);
    }
    printf ("\n");
}



int main ()
{
    int pid, child_pid;
    int size, i, status;
    printf ("Enter the number of Integers to Sort::::\t");
    scanf ("%d", &size);
    int a[size];
    int pArr[size];
    int cArr[size];
    for (i = 0; i < size; i++)
    {
        printf ("Enter number %d:", (i + 1));
        scanf ("%d", &a[i]);
```

tushar@tushar-virtual-machine: ~/OS Lab/week8

```
        pArr[i] = a[i];
        cArr[i] = a[i];
    }
    printf ("Your Entered Integers for Sorting\n");
    display (a, size);
    pid = getpid ();
    printf ("Current Process ID is : %d\n", pid);
    printf ("[ Forking Child Process ... ] \n");

    child_pid = fork ();
    if (child_pid < 0)
    {
        printf ("\nChild Process Creation Failed!!!!\n");
        exit (-1);
    }
    else if (child_pid == 0)
    {
            printf ("\nThe Child Process\n");
        printf ("\nchild process is %d", getpid ());
        printf ("\nparent of child process is %d", getppid ());
        printf ("Child is sorting the list of Integers by SELECTION SORT::\n");
        selectionSort (cArr, 0, size - 1);
        printf ("The sorted List by Child::\n");
        display (cArr, size);
        printf ("Child Process Completed ...\n");
        sleep (10);
        printf ("\nparent of child process is %d", getppid ());
    }
    else
    {
            printf("parent process %d started\n", getpid ());
        printf("Parent of parent is %d\n", getppid ());
        sleep(30);
        printf("The Parent Process\n");
        printf("Parent %d is sorting the list of Integers by INSERTION SORT\n", pid);

        partition (pArr, 0, size - 1);
        printf ("The sorted List by Parent::\n");
        display (pArr, size);
        wait (&status);
        printf ("Parent Process Completed ...\n");
    }
    return 0;
}
```

tushar@tushar-virtual-machine: ~/OS Lab/week8

```
        display (pArr, size);
        wait (&status);
        printf ("Parent Process Completed ...\n");
    }
    return 0;
}

int split (int a[], int lower, int upper)
{
    int i, p, q, t;
    p = lower + 1;
    q = upper;
    i = a[lower];
    while (q >= p)
    {
        while (a[p] < i)
            p++;

        while (a[q] > i)
            q--;

        if (q > p)
        {
            t = a[p];
            a[p] = a[q];
            a[q] = t;
        }
    }
    t = a[lower];
    a[lower] = a[q];
    a[q] = t;
    return q;
}

void selectionSort (int a[], int lower, int upper)
{
    int i;
    if (upper > lower)
    {
        i = split (a, lower, upper);
        selectionSort (a, lower, i - 1);
        selectionSort (a, i + 1, upper);
    }
}
```

Output:

```
tushar@tushar-virtual-machine: ~/OS Lab/week8
tushar@tushar-virtual-machine:~/OS Lab/week8$ vi week8Act2.c
tushar@tushar-virtual-machine:~/OS Lab/week8$ gcc week8Act2.c
week8Act2.c: In function ‘main’:
week8Act2.c:127:9: warning: implicit declaration of function ‘wait’ [-Wimplicit-function-declaration]
  127 |         wait (&status);
      |         ^~~~
tushar@tushar-virtual-machine:~/OS Lab/week8$ ./a.out
Enter the number of Integers to Sort::::        5
Enter number 1:20
Enter number 2:10
Enter number 3:30
Enter number 4:5
Enter number 5:50
Your Entered Integers for Sorting
20              10              30              5               50
Current Process ID is : 2071
[ Forking Child Process ... ]
parent process 2071 started
Parent of parent is 2050

The Child Process

child process is 2072
parent of child process is 2071Child is sorting the list of Integers by SELECTION SORT::
The sorted List by Child::
5               10              20              30              50
Child Process Completed ...

parent of child process is 2071The Parent Process
Parent 2071 is sorting the list of Integers by INSERTION SORT
The sorted List by Parent::
5               10              20              30              50
Parent Process Completed ...
tushar@tushar-virtual-machine:~/OS Lab/week8$
```

4. Write a program to print the Child process ID and Parent process ID in both Child and Parent processes.

Code :

```
tushar@tushar-virtual-machine: ~/OS Lab/week8
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
int main()
{
    int i;
    printf("hello before fork \n");
    printf("i : %d\n",i);
    i=fork();
    printf("\n");
    if(i==0)
    {
        printf("Child has started\n\n");
        printf("child printing first time \n");
        printf("getpid : %d getppid : %d \n",getpid(),getppid());
        sleep(5);
        printf("\nchild printing second time \n");
        printf("getpid : %d getppid : %d \n",getpid(),getppid());
    }
    else
    {
        printf("parent has started\n");
        printf("getpid : %d  getppid : %d \n",getpid(),getppid());
        printf("\n");
    }
    printf("Hi after fork i : %d\n",i);
    return 0;
}
```

Output:

```
tushar@tushar-virtual-machine: ~/OS Lab/week8
tushar@tushar-virtual-machine:~/OS Lab/week8$ vi week8Act4.c
tushar@tushar-virtual-machine:~/OS Lab/week8$ gcc week8Act4.c
tushar@tushar-virtual-machine:~/OS Lab/week8$ ./a.out
hello before fork
i : 21920

parent has started
getpid : 2182  getppid : 2050

Hi after fork i : 2183

Child has started

child printing first time
getpid : 2183 getppid : 2182
tushar@tushar-virtual-machine:~/OS Lab/week8$
child printing second time
getpid : 2183 getppid : 1003
Hi after fork i : 0

tushar@tushar-virtual-machine:~/OS Lab/week8$
```

9,10→Overlay Concepts

1->
example1.c

```
tushar@tushar-virtual-machine: ~/OS Lab/week9
#include<stdio.h>
#include<unistd.h>
int main()
{
    int p_id;
    p_id=getpid();
    printf("Process ID of example.c: %d\n",p_id);
    printf("\n");
    char *args[]={"./hello1", NULL};
    execv(args[0],args);
}
```

hello1.c

```
tushar@tushar-virtual-machine: ~/OS Lab/week9
#include<stdio.h>
#include<unistd.h>
int main()
{
    int p_id;
    p_id=getpid();
    printf("We are in hello.c\n");
    printf("Process ID of hello.c: %d\n",p_id);
    printf("\n");
}
```

Output:

```
tushar@tushar-virtual-machine:~/OS Lab/week9$ vi example1.c
tushar@tushar-virtual-machine:~/OS Lab/week9$ vi hello1.c
tushar@tushar-virtual-machine:~/OS Lab/week9$ gcc example1.c -o example1
tushar@tushar-virtual-machine:~/OS Lab/week9$ gcc hello1.c -o hello1
tushar@tushar-virtual-machine:~/OS Lab/week9$ ./example1
Process ID of example.c: 4194

We are in hello.c
Process ID of hello.c: 4194

tushar@tushar-virtual-machine:~/OS Lab/week9$
```

2->
example2.c

```
tushar@tushar-virtual-machine: ~/OS Lab/week9
#include<stdio.h>
#include<unistd.h>
#include<sys/types.h>
int main()
{
    int p_id,i;
    p_id=getpid();
    printf("Process ID of example.c: %d\n",p_id);
    printf("The control is in parent process\n");
    i=fork();
    if(i==0)
    {
        printf("The control is in child process\n");
        printf("Process ID of child: %d\n",getpid());
        printf("Calling hello.c from child\n");
        printf("\n");
        char *args[]={"./hello2", NULL};
        execv(args[0],args);
    }
}

~
~
:wq
```

hello2.c

```
tushar@tushar-virtual-machine: ~/OS Lab/week9
#include<stdio.h>
#include<unistd.h>
int main()
{
    int p_id;
    p_id=getpid();
    printf("We are in hello.c\n");
    printf("Process ID of hello.c: %d\n",p_id);
}
```

Output:

```
tushar@tushar-virtual-machine:~/OS Lab/week9$ vi example2.c
tushar@tushar-virtual-machine:~/OS Lab/week9$ vi hello2.c
tushar@tushar-virtual-machine:~/OS Lab/week9$ gcc example2.c -o example2
tushar@tushar-virtual-machine:~/OS Lab/week9$ gcc hello2.c -o hello2
tushar@tushar-virtual-machine:~/OS Lab/week9$ ./example2
Process ID of example.c: 4224
The control is in parent process
The control is in child process
Process ID of child: 4225
Calling hello.c from child

tushar@tushar-virtual-machine:~/OS Lab/week9$ We are in hello.c
Process ID of hello.c: 4225
```

11→Interprocess Communication Usiong Pipes
1->

```c
#include <stdio.h>
#include <unistd.h>

int main()
{
        int pipefds[2];
        int returnStatus;
        char writeMessages[2][20] = {"hi", "hello"};
        char readMessage[20];
        returnStatus = pipe(pipefds);
        if(returnStatus == -1)
        {
                printf("unable to create pipe\n");
                return 1;
        }
        printf("Writing to pipe - Message 1 id %s\n ", writeMessages[0]);
        write(pipefds[1], writeMessages[0], sizeof(writeMessages[0]));
        read(pipefds[0], readMessage, sizeof(readMessage));
        printf("Reading from pipe - Message 1 is %s\n", readMessage);
        printf("Writing to pipe - Message 2 is %s\n", writeMessages[1]);
        write(pipefds[1], writeMessages[1], sizeof(writeMessages[1]));
        read(pipefds[0], readMessage, sizeof(readMessage));
        printf("Reading from pipe - Message 2 is %s \n", readMessage);
        return 0;
}
~
```

```
tushar@tushar-virtual-machine:~$ vi week11Act1.c
tushar@tushar-virtual-machine:~$ gcc week11Act1.c
tushar@tushar-virtual-machine:~$ ./a.out
Writing to pipe - Message 1 id hi
 Reading from pipe - Message 1 is hi
Writing to pipe - Message 2 is hello
Reading from pipe - Message 2 is hello
```

2->

```
tushar@tusha
#include<stdio.h>
#include<unistd.h>

int main() {
   int pipefds[2];
   int returnstatus;
   int pid;
   char writemessages[2][20]={"Hi", "Hello"};
   char readmessage[20];
   returnstatus = pipe(pipefds);
   if (returnstatus == -1) {
      printf("Unable to create pipe\n");
      return 1;
   }
   pid = fork();

   // Child process
   if(pid == 0)
   {
      read(pipefds[0], readmessage, sizeof(readmessage));
      printf("Child Process - Reading from pipe - Message 1 is %s\n", readmessage);
      read(pipefds[0], readmessage, sizeof(readmessage));
      printf("Child Process - Reading from pipe - Message 2 is %s\n", readmessage);
   } else { //Parent process
      printf("Parent Process - Writing to pipe - Message 1 is %s\n", writemessages[0]);
      write(pipefds[1], writemessages[0], sizeof(writemessages[0]));
      printf("Parent Process - Writing to pipe - Message 2 is %s\n", writemessages[1]);
      write(pipefds[1], writemessages[1], sizeof(writemessages[1]));
   }
   return 0;
}
~
```

```
tushar@tushar-virtual-machine:~$ vi week11Act2.c
tushar@tushar-virtual-machine:~$ gcc week11Act2.c
tushar@tushar-virtual-machine:~$ ./a.out
Parent Process - Writing to pipe - Message 1 is Hi
Parent Process - Writing to pipe - Message 2 is Hello
Child Process - Reading from pipe - Message 1 is Hi
Child Process - Reading from pipe - Message 2 is Hello
tushar@tushar-virtual-machine:~$
```

3->

tushar@tushar-virtual-machine: ~

```c
#include<stdio.h>
#include<unistd.h>

int main() {
   int pipefds1[2], pipefds2[2];
   int returnstatus1, returnstatus2;
   int pid;
   char pipe1writemessage[20] = "Hi";
   char pipe2writemessage[20] = "Hello";
   char readmessage[20];
   returnstatus1 = pipe(pipefds1);

   if (returnstatus1 == -1) {
      printf("Unable to create pipe 1 \n");
      return 1;
   }
   returnstatus2 = pipe(pipefds2);

   if (returnstatus2 == -1) {
      printf("Unable to create pipe 2 \n");
      return 1;
   }
   pid = fork();

    if(pid != 0)
    {
      close(pipefds1[0]); // Close the unwanted pipe1 read side
      close(pipefds2[1]); // Close the unwanted pipe2 write side
      printf("In Parent: Writing to pipe 1 - Message is %s\n", pipe1writemessage);
      write(pipefds1[1], pipe1writemessage, sizeof(pipe1writemessage));
      read(pipefds2[0], readmessage, sizeof(readmessage));
      printf("In Parent: Reading from pipe 2 - Message is %s\n", readmessage);
   }
   else { //child process
      close(pipefds1[1]); // Close the unwanted pipe1 write side
      close(pipefds2[0]); // Close the unwanted pipe2 read side
      read(pipefds1[0], readmessage, sizeof(readmessage));
      printf("In Child: Reading from pipe 1 - Message is %s\n", readmessage);
      printf("In Child: Writing to pipe 2 - Message is %s\n", pipe2writemessage);
      write(pipefds2[1], pipe2writemessage, sizeof(pipe2writemessage));
   }
   return 0;
}
~
"tempWeek11Act3.c" 43 lines, 1449 characters
```

```
tushar@tushar-virtual-machine:~$ vi tempWeek11Act3.c
tushar@tushar-virtual-machine:~$ gcc tempWeek11Act3.c
tushar@tushar-virtual-machine:~$ ./a.out
In Parent: Writing to pipe 1 - Message is Hi
In Child: Reading from pipe 1 - Message is Hi
In Child: Writing to pipe 2 - Message is Hello
In Parent: Reading from pipe 2 - Message is Hello
tushar@tushar-virtual-machine:~$
```

12→Readers Writers Problem

12)a.
Writer

tushar@tushar-virtual-machine: ~/OS Lab/week12

```
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
#include <unistd.h>
int main()
{
        key_t key = ftok("shmfile",65);
        int shmid = shmget(key,1024,0666|IPC_CREAT);
        char *str = (char*) shmat(shmid,(void*)0,0);
        printf("Write Data : ");
        scanf("%s",str);
        printf("Data written in memory: %s\n",str);
        shmdt(str);
        return 0;
}
```

Output

```
tushar@tushar-virtual-machine:~/OS Lab/week12$ vi week12Act1writer.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ gcc week12Act1writer.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ ./a.out
Write Data : Dummy
Data written in memory: Dummy
tushar@tushar-virtual-machine:~/OS Lab/week12$
```

Reader

tushar@tushar-virtual-machine: ~/OS Lab/week12

```
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/shm.h>
int main()
{
        key_t key=ftok("shmfile",65);
        int shmid=shmget(key,1024,0666|IPC_CREAT);
        char *str=(char*) shmat(shmid,(void*)0,0);
        printf("Data read from memory: %s\n",str);
        shmdt(str);
        shmctl(shmid,IPC_RMID,NULL);
        return 0;
}
```

Output

```
tushar@tushar-virtual-machine:~/OS Lab/week12$ vi week12Act1reader.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ gcc week12Act1reader.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ ./a.out
Data read from memory: Dummy
tushar@tushar-virtual-machine:~/OS Lab/week12$
```

12)b.

Writer

```
tushar@tushar-virtual-machine: ~/OS Lab/week12
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#define MAX 10

struct mesg_buffer {
        long mesg_type;
        char mesg_text[100];
} message;
int main()
{
        key_t key;
        int msgid;

        key = ftok("progfile",65);
        msgid = msgget(key, 0666 | IPC_CREAT);
        message.mesg_type = 1;
        printf("Write Data : ");
        fgets(message.mesg_text,MAX,stdin);
        msgsnd(msgid, &message, sizeof(message), 0);
        printf("Data send is : %s \n", message.mesg_text);
        return 0;
}
"week12Act2writer.c" 23 lines, 536 characters
```

Output

```
tushar@tushar-virtual-machine:~/OS Lab/week12$ vi week12Act2writer.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ gcc week12Act2writer.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ ./a.out
Write Data : Dummy Data
Data send is : Dummy Dat
tushar@tushar-virtual-machine:~/OS Lab/week12$
```

Reader

```
tushar@tushar-virtual-machine: ~/OS Lab/week12
#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
struct mesg_buffer {
        long mesg_type;
        char mesg_text[100];
} message;
int main()
{
        key_t key;
        int msgid;
        key = ftok("progfile",65);
        msgid = msgget(key, 0666 | IPC_CREAT);
        msgrcv(msgid, &message, sizeof(message), 1, 0);
        printf("Data Received is : %s\n",message.mesg_text);
        msgctl(msgid, IPC_RMID, NULL);
        return 0;
}
```

Output

```
tushar@tushar-virtual-machine:~/OS Lab/week12$ vi week12Act2reader.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ gcc week12Act2reader.c
tushar@tushar-virtual-machine:~/OS Lab/week12$ ./a.out
Data Received is : Dummy Dat
tushar@tushar-virtual-machine:~/OS Lab/week12$
```

13→ Process Synchronization

1) Producer consumer problem using semaphore

tushar@tushar-virtual-machine: ~/OS Lab/week13

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
int mutex=1;
int full=0;
int empty=10,x=0;
void producer()
{
    --mutex;
    ++full;
    --empty;
    x++;
    printf("Producer produces items: %d",x);
}
void consumer()
{
    --mutex;
    --full;
    ++empty;
    printf("Consumer consumes items: %d",x);
    x--;
    ++mutex;
}
int main()
{
    int n,i;
    printf("\n1. Press 1 for Producer\n2. Press 2 for consumer\n3. Press 3 for exit");
    for(int i=1;i>0;i++)
    {
        printf("\nEnter your choice : ");
        scanf("%d",&n);
        switch(n)
        {
            case 1:
                if((mutex==1) && (empty!=0))
                {
                    producer();
                }
                else
                {
                    printf("Buffer is full");
                }
                break;
```

```
            case 2:
                if((mutex==1)&&(full!=0))
                {
                    consumer();
                }
                else
                {
                    printf("Buffer is empty");
                }
                break;
            case 3:
                exit(0);
                break;
        }
    }
    return 0;
}
```

Output:

```
tushar@tushar-virtual-machine:~/OS Lab/week13$ vi week13Act2.c
tushar@tushar-virtual-machine:~/OS Lab/week13$ gcc week13Act2.c
tushar@tushar-virtual-machine:~/OS Lab/week13$ ./a.out

1. Press 1 for Producer
2. Press 2 for consumer
3. Press 3 for exit
Enter your choice : 1
Producer produces items: 1
Enter your choice : 2
Buffer is empty
Enter your choice : 3
tushar@tushar-virtual-machine:~/OS Lab/week13$
```

2) Mutual Exclusion

tushar@tushar-virtual-machine: ~/OS Lab/week13

```
#include<unistd.h>
#include<stdio.h>
#include<sys/wait.h>
#include<sys/ipc.h>
#include<sys/sem.h>
int main()
{
    int pid,semid,val;
    struct sembuf sop;
    semid=semget((key_t)6,1,IPC_CREAT|0666);
    pid=fork();
    sop.sem_num=0;
    sop.sem_op=0;
    sop.sem_flg=0;
    if (pid!=0)
    {
        sleep(1);
        printf("The Parent waits for WAIT signal\n");
        semop(semid,&sop,1);
        printf("The Parent WAKED UP & doing her job\n");
        sleep(10);
        printf("Parent Over\n");
    }
    else
    {
        printf("The Child sets WAIT signal & doing her job\n");
        semctl(semid,0,SETVAL,1);
        sleep(10);
        printf("The Child sets WAKE signal & finished her job\n");
        semctl(semid,0,SETVAL,0);
        printf("Child Over\n");
    }
    return 0;
}
~
~
~
~
~
~
~
~
```

Output:

```
tushar@tushar-virtual-machine:~/OS Lab/week13$ vi week13Act1.c
tushar@tushar-virtual-machine:~/OS Lab/week13$ gcc week13Act1.c
tushar@tushar-virtual-machine:~/OS Lab/week13$ ./a.out
The Child sets WAIT signal & doing her job
The Parent waits for WAIT signal
The Child sets WAKE signal & finished her job
Child Over
The Parent WAKED UP & doing her job
Parent Over
tushar@tushar-virtual-machine:~/OS Lab/week13$
```