Filters

Enter Data Same as Below(Using Vi Command)

```
1;John;Los_Angeles;California;29;Aug;Male;174;96;4
2;Max;Chicago;Illinois;45;Jan;Male;189;87;2
3;Samantha;Houston;Texas;25;Jul;Female;185;110;4
4;Penelope;Phoenix;Arizona;48;Sep;Female;195;104;3
5;Henry;Fort_Worth;Texas;35;Apr;Male;149;61;3
6;William;San_Francisco;California;37;Apr;Male;189;104;3
7;Scott;Las_Vegas;Nevada;39;May;Male;147;92;5
8;James;Miami;Florida;26;Oct;Male;154;111;5
9;Jason;Oakland;California;40;Feb;Male;174;90;3
10;Megan;Cincinnati;Ohio;24;Nov;Female;169;103;4
```

```
Q1.  sort -t";" -k3 table
Q2.  sort -t";" -k4 table
Q3.  sort -t";" -k10 table
Q4.  sort -t";" -k33 -k44 table
Q5.  cut -d";" -f1,2,5,7,10 table
Q6.  Cut -d”;” -f2 table | tr a-z A-Z
Q7.  sort -t";" -k5 table
Q8.  sort -t";" -M -k6 table > Personnel_data_month2022.dat
Q9. cut -d”;” -f2f table | grep -e S -e J > SJ_Names.txt
Q10. Cut -d”;” -f2,3,4,5 table | sort ‘t”;” -k1 > capital
```

Q1.

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/wait.h>
```

```
#include<unistd.h>

int main()
{
    pid_t cpid;
    if (fork()== 0)
        exit(0);
    else
        cpid = wait(NULL);
    printf("Parent pid = %d\n", getpid());
    printf("Child pid = %d\n", cpid);
    return 0;
}
```

Q2.

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

int main()
{
// fork() Create a child process
int pid = fork();
if (pid > 0) {
    printf("I am Parent process\n");
    printf("ID : %d\n\n", getpid());}
else if (pid == 0) {
    printf("I am Child process\n");
// getpid() will return process id of child process
    printf("ID: %d\n", getpid());  }
else {
       printf("Failed to create child process");
}
return 0;
}
```

Q3.

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

int main (){
  int num[20];
  int i, j, a, status;
  printf("Enter the elements\n");
  for (i = 0; i < 5; ++i)
     scanf("%d", &num[i]);
     pid_t pid;
pid=fork();
if(pid==0)
{
printf("In child process \n");
  for (i = 0; i < 5; ++i){
     for (j = i + 1; j < 5; ++j){
        if (num[i] > num[j]){
           a = num[i];
           num[i] = num[j];
        num[j] = a;
      }
    }
  }
  printf("The numbers in ascending order is:\n");
  for (i = 0; i < 5; ++i){
    printf("%d\n", num[i]);
  }
}

else if(pid>0)
{
wait(&status);
if(WIFEXITED (status))
printf("Child returned");
printf("In parents procees\n");
for (i = 0; i < 5; ++i)
    printf("%d\n", num[i]);}
    return 0;}
```

Q4.

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

int main()
```

```
{
   pid_t pid = fork();

  switch(pid)
   {
      case -1:
         printf("Error in creating child process\n");
         break;

      case 0:
         sleep(5);
         printf("Child Process executing\n");
         printf("Process id:%d\n",getpid());
         break;

      default:
         printf("Parent Process executing\n");
         printf("Process id:%d\n",getpid());
         exit(0);
   }
   return 0;
}
```

Q5.

```
#include<stdio.h>
#include<dirent.h>
main()
{
char dirname[10];
DIR*p;
struct dirent *d;
printf("Enter directory name\n");
scanf("%s",dirname);
p=opendir(dirname);
if(p==NULL)
  {
  perror("Cannot find directory");
  exit(-1);
  }
while(d=readdir(p))
  printf("%s\n",d->d_name);
}
```