1. Get three values x, y, z and write a program to print 1 if x is the middle value, 2 if y is the middle value and 3 if z is the middle value. Assume that all three variables (x, y, z) are distinct and have different values.

CODE :

```
#include<stdio.h>

int main(){

            int x,y,z;

            printf("Enter x value : ");

            scanf("%d",&x);

            printf("Enter y value : ");

            scanf("%d",&y);

            printf("Enter z value : ");

            scanf("%d",&z);

             if ((x > y && x < z) || (x < y && x > z)) {

       printf("1 ");

   } else if ((y > x && y < z) || (y < x && y > z)) {

       printf("2");

   } else {

       printf("3");

   }

   return 0;

}
```

2. A password is said to be strong if it satisfies the following criteria:

It contains at least one lowercase English character.

It contains at least one uppercase English character.

It contains at least one special character.

The special characters are: !@#$%^&*()-+

Its length is at least 8.

It contains at least one digit. Given a string, find its strength.

CODE :

```
#include<stdio.h>
#include<ctype.h>
#include<string.h>
int main(){
 char password[20];
 int i;
  int upper=0,lower=0,special=0,digit=0;
 printf("Enter the password : ");
 scanf("%s",password);
  for(i=0; password[i] != '\0'; i++){
      if(password[i]>='A' && password[i]<='Z') {
        upper++;
  }
  else if(password[i]>='a' && password[i]<='z') {
        lower++;
  }
  else if((password[i]=='!') || (password[i]=='@') ||(password[i]=='#') ||
(password[i]=='$') ||(password[i]=='%') ||
(password[i]=='^')||(password[i]=='&')||(password[i]=='(')||(password[i]==')')||(passw
ord[i]=='+')||(password[i]=='-')){
    special++;
  }
  else if (password[i] >= '0' && password[i] <= '9') {
      digit++;


  }
}

        if(upper<1 || lower<1 || special<1||strlen(password)<8||digit<1 ){

                printf("password is weak");

                }

                else{

                        printf("password is  strong");

                }


  return 0;

}
```

3. A firm creates projects for which a certain number of hours are needed. The firm has a certain number of days. During 10% of the days, the workers are being trained and cannot work on the project. A normal working day is 8 hours long. The project is important for the firm and every worker must work on it with overtime of 2 hours per day. The hours must be rounded down to the nearest integer (for example, 6.98 hours are rounded to 6 hours). Write a program that calculates whether the firm can finish the project on time and how many hours more are needed or left.

CODE :

```
#include<stdio.h>

#include<math.h>

int main(){

        int number_of_days,number_of_workers,singleDay_hours=8,actualDaysToWork,calculatedWorkHoursByWorkers;

        double tenPercent = 100 * 0.1;

  float total_number_of_hours_needed;

  int RoundOff_hours;

        printf("Enter the total number of hours needed : ");

        scanf("%f",&total_number_of_hours_needed);

        RoundOff_hours=round(total_number_of_hours_needed);

        printf("\nEnter the number of days : ");
```

```
            scanf("%d",&number_of_days);

            actualDaysToWork=number_of_days-tenPercent;

            printf("\nEnter the number of workers : ");

            scanf("%d",&number_of_workers);

            calculatedWorkHoursByWorkers=(2*actualDaysToWork)*number_of_workers;


if (calculatedWorkHoursByWorkers >= RoundOff_hours) {

       printf("Yes! %d hours left.\n", calculatedWorkHoursByWorkers - RoundOff_hours);

    } else {

       printf("Not enough time! %d hours needed.\n", RoundOff_hours -
calculatedWorkHoursByWorkers);

    }

            return 0;

}
```