```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main()
{
   int n;
   scanf("%d",&n);
   char nayan[30];
   fgets(nayan,30, stdin);
   int p = strlen(nayan);
   if (p > 0 && nayan[p - 1] == '\n')
   nayan[p-1] = '\0';

   for(int i=0;i<n;i++){
      fgets(nayan,30, stdin);
      p = strlen(nayan);
      if (p > 0 && nayan[p - 1] == '\n')
      nayan[p-1] = '\0';

      if(strlen(nayan)<3){
      if(nayan[0] == '+' && nayan[1] == '\0'){
         printf("Arithmatic Addition Operator \n");
      }
      else if(nayan[0] == '-' && nayan[1] == '\0'){
         printf("Arithmatic Substraction Operator \n");
      }
      else if(nayan[0] == '*' && nayan[1] == '\0'){
         printf("Arithmatic Multiplication Operator \n");
```

```
}
else if(nayan[0] == '/' && nayan[1] == '\0'){
   printf("Arithmatic Substraction Operator \n");
}
else if(nayan[0] == '%' && nayan[1] == '\0'){
   printf("Arithmatic Modulus Operator \n");
}
else if(nayan[0] == '+' && nayan[1] == '+'){
   printf("Increment Operator \n");
}
else if(nayan[0] == '-' && nayan[1] == '-'){
   printf("Decrement Operator \n");
}
else if(nayan[0] == '=' && nayan[1] == '\0'){
   printf("Assignment Operator \n");
}
else if(nayan[0] == '>' && nayan[1] == '\0'){
   printf("Relational Greater than Operator \n");
}
else if(nayan[0] == '>' && nayan[1] == '='){
   printf("Relational Greater than equal Operator \n");
}
else if(nayan[0] == '<' && nayan[1] == '\0'){
   printf("Relational less than Operator \n");
}
else if(nayan[0] == '<' && nayan[1] == '='){
   printf("Relational Greater than equal Operator \n");
}
else if(nayan[0] == '=' && nayan[1] == '='){
```

```
    printf("Relational equal to Operator \n");
}
else if(nayan[0] == '!' && nayan[1] == '='){
    printf("Relational not equal to Operator \n");
}
else if(nayan[0] == '&' && nayan[1] == '&'){
    printf("Logical and Operator \n");
}
else if(nayan[0] == '|' && nayan[1] == '|'){
    printf("Logical or Operator \n");
}
else if(nayan[0] == '!' && nayan[1] == '\0'){
    printf("Logical not Operator \n");
}
else if(nayan[0] == '&' && nayan[1] == '\0'){
    printf("Bitwise and Operator \n");
}
else if(nayan[0] == '|' && nayan[1] == '\0'){
    printf("Bitwise or Operator \n");
}
else if(nayan[0] == '~' && nayan[1] == '\0'){
    printf("Bitwise compliment Operator \n");
}
else if(nayan[0] == '^' && nayan[1] == '\0'){
    printf("Bitwise XOR Operator \n");
}
else if(nayan[0] == '>' && nayan[1] == '>'){
    printf("Bitwise right shift Operator \n");
}
```

```
        else if(nayan[0] == '<' && nayan[1] == '<'){
            printf("Bitwise left shift Operator \n");
        }

        else{
            printf("This is not an operator \n");
        }

        }
        else{
             printf("This is not an operator \n");
        }


}
}
```