MAJOR ASSIGNMENT

1. Write a program to model a simple calculator. Each data line should consist of the next operation to be performed from the list below and the right operand. Assume the left operand is the accumulator value (initial value of 0). You need a function scan data with two output parameters that returns the operator and right operand scanned from a data line. You need a function do next op that performs the required operation. do next op has two input parameters (the operator and operand) and one input/output parameter (the accumulator). The valid operators are:
+ add− subtract∗ multiply/ divide^ power (raise left operand to power of right operand)
q quit
Your calculator should display the accumulator value after each operation. A sample run follows.
+ 5.0 result so far is 5.0 ^ 2 result so far is 25.0 / 2.0 result so far is 12.5 q 0 final result is 12.5

**PROBLEM:** Simple Calculator

**IMPLEMENTATION:**

```
#include<stdio.h>
#include<math.h>
void scan_data(char *operator, double *operand) {
   printf("Enter operation and operand: ");
   scanf(" %c %lf", operator, operand);
}
void do_next_op(char operator, double operand, double *accumulator) {
   switch (operator) {
      case '+': *accumulator += operand; break;
      case '-': *accumulator -= operand; break;
      case '*': *accumulator *= operand; break;
      case '/':
         if (operand != 0)
            *accumulator /= operand;
         else
            printf("Error: Division by zero.\n");
         break;
      case '^':
         *accumulator = pow(*accumulator, operand);
         break;
      default:
         printf("Invalid operator.\n");
   }
}
int main() {
   char operator;
   double operand, accumulator = 0.0;
   while (1) {
      scan_data(&operator, &operand);
      if (operator == 'q') break;
      do_next_op(operator, operand, &accumulator);
      printf("Result so far is %.2lf\n", accumulator);
   }
   printf("Final result is %.2lf\n", accumulator);
   return 0;
}
```

```
Enter operation and operand: + 10
Result so far is 10.00
Enter operation and operand: * 2
Result so far is 20.00
Enter operation and operand: - 5
Result so far is 15.00
Enter operation and operand: / 3
Result so far is 5.00
```

Enter operation and operand: q 0
Final result is 5.00

2. The table below shows the normal boiling points of several substances. Write a program that prompts the user for the observed boiling point of a substance in °C and identifies the substance if the observed boiling point is within 5% of the expected boiling point. If the data input is more than 5% higher or lower than any of the boiling points in the table, the program should output the message Substance Unknown.

| Substance | Normal boiling point (°C) |
|---|---|
| Water | 100 |
| Mercury | 357 |
| Copper | 1187 |
| Silver | 2193 |
| Gold | 2660 |

Your program should define and call a function within x percent that takes as parameters a reference value ref , a data value data , and a percentage value x and returns 1 meaning true if data is within x % of ref —that is, (ref – x% * ref) ≤ data ≤ (ref + x % * ref) . Otherwise within x percent would return zero, meaning false. For example, the call within x percent(357, 323, 10) would return true, since 10% of 357 is 35.7, and 323 falls between 321.3 and 392.7.

**PROBLEM:** Identify Substance Based on Boiling Point

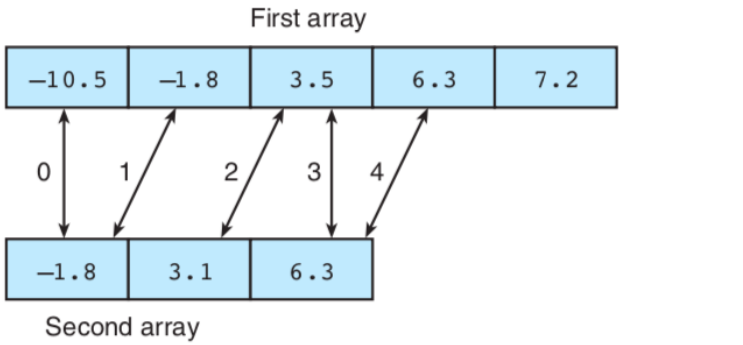
**IMPLEMENTATION:**

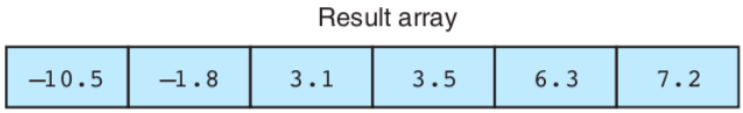
```
#include<stdio.h>
int within_x_percent(double ref, double data, double x) {
   double margin = ref * (x / 100.0);
   return (ref - margin <= data && data <= ref + margin);
}

const char* identify_substance(double boiling_point) {
   struct {
      const char* name;
      double bp;
   } substances[] = {
      {"Water", 100},
      {"Mercury", 357},
      {"Copper", 1187},
      {"Silver", 2193},
      {"Gold", 2660}
   };
   for (int i = 0; i < 5; i++) {
      if (within_x_percent(substances[i].bp, boiling_point, 5.0)) {
         return substances[i].name;
      }
   }
   return "Substance Unknown";
}
int main() {
   double observed_bp;
   printf("Enter the observed boiling point (°C): ");
   scanf("%lf", &observed_bp);
   printf("Substance: %s\n", identify_substance(observed_bp));
   return 0;
}
```

Enter the observed boiling point (°C): 102
Substance: Water

3. Write a function that will merge the contents of two sorted (ascending order) arrays of type double values, storing the result in an array output parameter (still in ascending order). The function should not assume that both its input parameter arrays are the same length but can assume that one array does not contain two copies of the same value. The result array should also contain no duplicate values.




**PROBLEM:** Merge Two Sorted Arrays

**IMPLEMENTATION:**

```
#include<stdio.h>
void merge_sorted_arrays(double array1[], int size1, double array2[], int size2, double result[], int *result_size)
{
   int i = 0, j = 0, k = 0;
   while (i < size1 && j < size2) {
      if (array1[i] < array2[j]) {
         result[k++] = array1[i++];
      } else if (array1[i] > array2[j]) {
         result[k++] = array2[j++];
      } else {
         result[k++] = array1[i++];
         j++;
      }
   }
   while (i < size1) {
      result[k++] = array1[i++];
   }
   while (j < size2) {
      result[k++] = array2[j++];
   }
   *result_size = k;
}
int main() {
   double array1[] = {-10.5, -1.8, 3.5, 6.3, 7.2};
   double array2[] = {-1.8, 3.1, 6.3};
   double result[10];
   int result_size;
   merge_sorted_arrays(array1, 5, array2, 3, result, &result_size);
   printf("Merged array: ");
   for (int i = 0; i < result_size; i++) {
      printf("%.1f ", result[i]);
   }
   printf("\n");
   return 0;
}
```

Array 1: {-10.5, -1.8, 3.5, 6.3, 7.2}
Array 2: {-1.8, 3.1, 6.3}
Merged array: -10.5 -1.8 3.1 3.5 6.3 7.2