CS2351 Data Structures

Insertion Sort, Selection Sort, and Merge Sort

Insertion Sort

```
#include <stdio.h>

int main() {
  int B[8] = { 18, 3, 5, 1, 12, 17, 8, 2 }, i, j, temp;

  for ( i = 0; i < 8; i++ )
    for ( j = i-1; j >= 0; j-- )
       if ( B[j] > B[j+1] )
       {  temp = B[j];  B[j] = B[j+1];  B[j+1] = temp; }
       else break;

  for ( i = 0; i < 8; i++ )
    printf("%d\n", B[i]);

  return 0;
}
```

Selection Sort

```
#include <stdio.h>

int main() {
  int B[8] = { 18, 3, 5, 1, 12, 17, 8, 2 }, i, j, temp, min_pos;

  for ( i = 0; i < 8; i++ ) {
    min_pos = i;
    for ( j = i+1; j < 8; j++ )
       if ( B[j] < B[min_pos] ) min_pos = j;

    temp = B[i];  B[i] = B[min_pos];  B[min_pos] = temp;
  }
  for ( i = 0; i < 8; i++ )
    printf("%d\n", B[i]);

  return 0;
}
```

Merge Sort

```
#include <stdio.h>

void merge(int *A, int i, int j, int m) {
  int p, q, r, *C;

  C = (int *) malloc( (j-i+1) * sizeof(int) );

  p = i; q = m+1; r = 0;
  while ( p <= m && q <= j ) {
     if ( A[p] < A[q] )  {  C[r] = A[p];  r++;  p++;  }
     else                {  C[r] = A[q];  r++;  q++;  }
  }

  if ( p > m )
    while ( q <= j )
    {  C[r] = A[q];  r++;  q++;  }
  else
    while ( p <= m )
    {  C[r] = A[p];  r++;  p++;  }

  for ( p = i; p <= j; p++ )
    A[p] = C[p-i];

  free(C);
}

void mergesort(int *A, int i, int j) {
  int m = (i+j)/2 ;
  if ( i >= j ) return;
  mergesort(A, i, m);
  mergesort(A, m+1, j);
  merge(A, i, j, m);
}

int main() {
  int B[8] = { 18, 3, 5, 1, 12, 17, 8, 2 }, i;

  mergesort(B, 0, 7);

  for ( i = 0; i < 8; i++ )
    printf("%d\n", B[i]);

  return 0;
}
```