Utilize a ideia do ponteiro para função pela função qsort() para implementar sua própria função de ordenação. Para isso, sua função deverá receber, entre outros argumentos, um ponteiro para a função de comparação.

Programa (Input):

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


float compare(float *ordem2, int n){
    float aux;
    for(int i = 0; i < n; i++){
        for(int j = i; j < n; j++){
            if(ordem2[i] > ordem2[j]){
                aux = ordem2[i];
                ordem2[i] = ordem2[j];
                ordem2[j] = aux;
            }
        }
    }
    return *ordem2;
}

float crescente(float (*px)(float *, int), float *ordem2){

  int n;
    printf("Digite o número de elementos que deseja ordenar: ");
    scanf ("%i", &n);
    ordem2 = (float*) malloc(n*sizeof(float));
    printf("\n");

        for(int i = 0; i < n; i++){
            printf("Digite o elemento %d: \n", i+1);
            scanf("%f", &ordem2[i]);
        }
    px(ordem2, n);

    printf("A ordem crescente dos valores informados segue abaixo: \n");
    for(int i =0; i < n; i++){
            printf("%f  ", ordem2[i]);
    }
    free(ordem2);
    return 0;
}

int main () {
    float *ordem1;
    crescente(compare, ordem1);
```

```
    return 0;
}
```

Output (Exemplo):