# Exercise 1. Answer Sheet

Student's Name: Tsuyoshi Kumamoto Student's ID:s1250050

***Problem 1.*** *(30 points)* For each function ***f(n)*** and time ***T*** in the following table, determine the largest size *n* of a problem that can be solved in time ***T***, assuming that the algorithm to solve the problem takes ***f(n)*** milliseconds.

| ***f(n)*** | ***T*** = 1 second | ***T*** = 1 minute | ***T*** = 1 hour | ***T*** = 1 day | ***T*** = 1 month (30 days) |
|---|---|---|---|---|---|
| $\sqrt{n}$ | 31.62 | 244.95 | 6000 | 9295.16 | 50911.69 |
| $n$ | 1000 | 60000 | 36000000 | 86400000 | 2592000000 |
| $n^2$ | 1000000 | $3.6 \times 10^9$ | $1.296 \times 10^{15}$ | $7.46496 \times 10^{15}$ | $6.718464 \times 10^{18}$ |
| $n^3$ | $1.0 \times 10^9$ | $2.16 \times 10^{14}$ | $4.6656 \times 10^{22}$ | $6.44972544 \times 10^{23}$ | $1.7414258688 \times 10^{28}$ |
| $2^n$ | $2^{1000}$ | $2^{60000}$ | $2^{36000000}$ | $2^{86400000}$ | $2^{2592000000}$ |

***Problem 2.*** *(30 points)* Consider sorting ***n*** numbers stored in array ***A*** by first finding the smallest element of ***A*** and exchanging it with the first element of the array, i.e. ***A[1]***. Them find the second smallest element of ***A***, and exchange it with ***A[2]***. Continue in this manner for the first *n-1* elements of *A*.

a) Write a pseudo-code for this algorithm which is known as "**Selection Sort**".

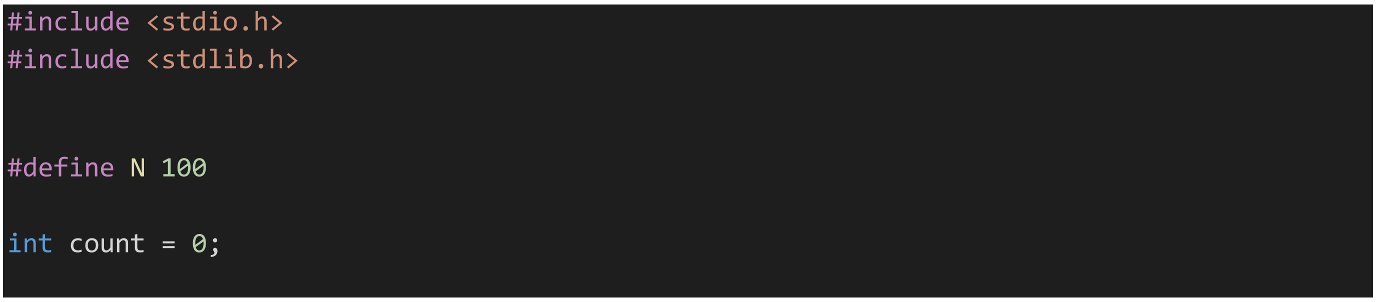
```
for i=0 to A.length-1
    minimam = i
    for j=i to A.length-1
        if A[j]<A[i]
            minimam=j
        A[i]=A[j
```

b) What is the time complexity of the Selection Sort algorithm?

The computation time of the selection sort is O $(n^2)$ in the worst case, so it is late in time.

***Problem 3.*** *(40 points)* Using the pseudo-code for **Merge Sort** algorithm given at the lecture, write a program implementing the **Merge Sort** algorithm. Use any programming language you know. Upload your source code with instructions how to compile/run it. Give the input data and the program output in the space below.

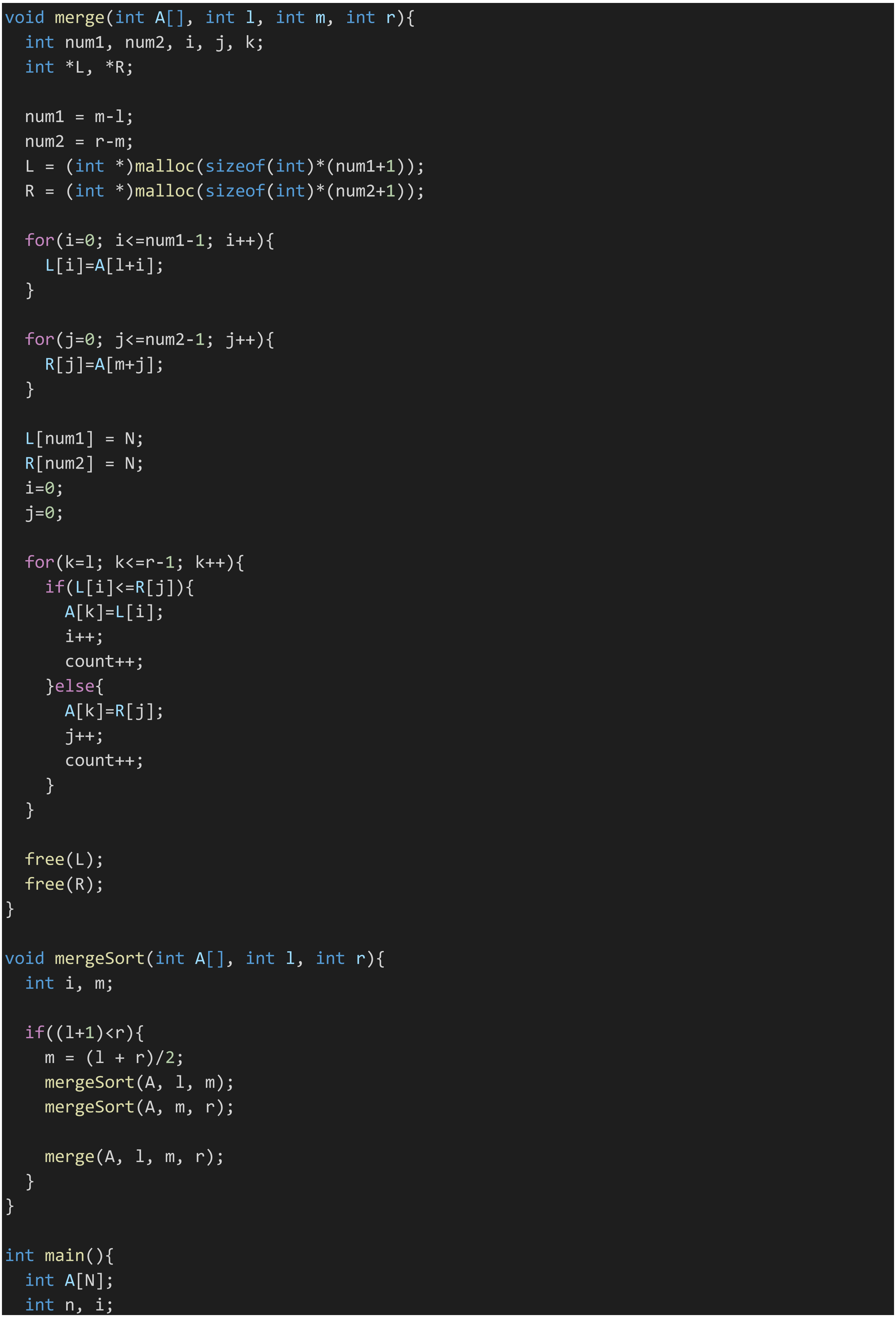
Put your answer here.



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


#define N 100

int count = 0;
```



```
void merge(int A[], int l, int m, int r){
  int num1, num2, i, j, k;
  int *L, *R;

  num1 = m-l;
  num2 = r-m;
  L = (int *)malloc(sizeof(int)*(num1+1));
  R = (int *)malloc(sizeof(int)*(num2+1));

  for(i=0; i<=num1-1; i++){
    L[i]=A[l+i];
  }

  for(j=0; j<=num2-1; j++){
    R[j]=A[m+j];
  }

  L[num1] = N;
  R[num2] = N;
  i=0;
  j=0;

  for(k=l; k<=r-1; k++){
    if(L[i]<=R[j]){
      A[k]=L[i];
      i++;
      count++;
    }else{
      A[k]=R[j];
      j++;
      count++;
    }
  }

  free(L);
  free(R);
}

void mergeSort(int A[], int l, int r){
  int i, m;

  if((l+1)<r){
    m = (l + r)/2;
    mergeSort(A, l, m);
    mergeSort(A, m, r);

    merge(A, l, m, r);
  }
}

int main(){
  int A[N];
  int n, i;
  scanf("%d",&n);

  for(i=0; i<n; i++){
    scanf("%d",&A[i]);
  }

  mergeSort(A, 0 ,n);

  for(i=0; i<n; i++){
    printf("%d",A[i]);
    if(i<n-1){
      printf(" ");
    }
  }

  printf("\n");

  return 0;
}
```

```
10
12
6
5
4
2
8
7
9
3
2
2 2 3 4 5 6 7 8 9 12
```