

Question1

A: The code of this question is as following:

```
#include<iostream>
#include<stdlib.h>
#include<ctime>
using namespace std;

int partition(int A[],int p,int q)
{
    int d,i,j,x;
    x = A[p];
    i = p;
    for (j = p + 1; j <= q; j++)
    {
        d = 0;
        if (A[j] <= x)
        {
            i++;
            d = A[j];
            A[j] = A[i];
            A[i] = d;
        }
    }
    d = A[i];
    A[i] = A[p];
    A[p] = d;
    return i;
}

int randomized_partition(int A[], int p, int q)
{
    int i,d,k;
    i = (int)((double)rand()/RAND_MAX)*(q-p)+p;
    d = A[i];
    A[i] = A[p];
    A[p] = d;
    k = partition(A, p, q);
    return k;
}

void randomized_quicksort(int A[], int p, int q)
{

```

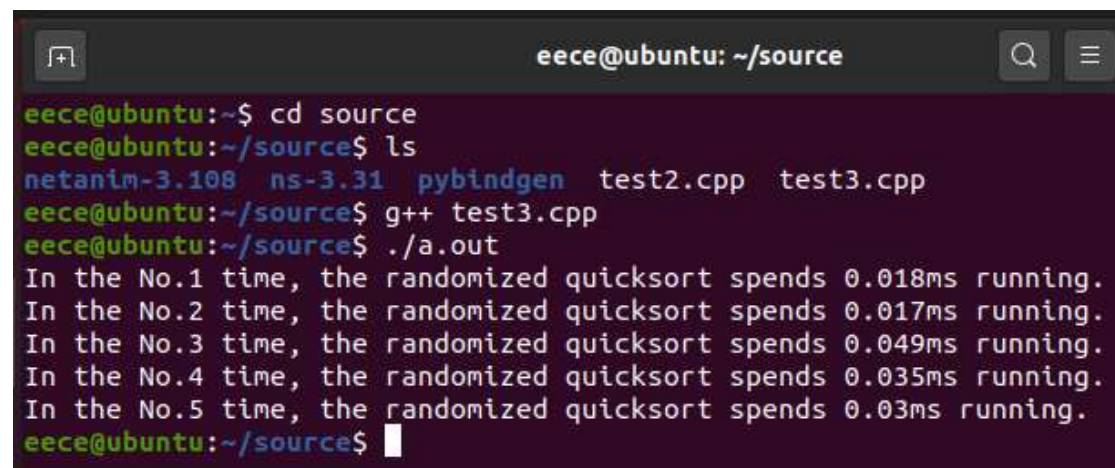
```

int r;
if (p < q)
{
    r = randomized_partition(A, p, q);
    randomized_quicksort(A, p, r - 1);
    randomized_quicksort(A, r + 1, q);
}
}

int main()
{
    int A[101];
    int i,d;
    double n;
    clock_t start, end;
    for (i = 1; i <=100; i++)
    {
        A[i] = i ;
    }
    for(i=1;i<=5;i++)
    {
        start=clock();
        randomized_quicksort(A, 1, 100);
        end=clock();
        n= (double)(end - start) / (double)(CLOCKS_PER_SEC)*1000;
        cout<<"In the No."<<i<<" time, the randomized quicksort spends "<<n<<"ms running.";
        cout<<endl;
    }
}

```

And the result of the codes is shown as the picture below:



The image shows a terminal window with the following commands and output:

```

eece@ubuntu: ~/source
eece@ubuntu:~$ cd source
eece@ubuntu:~/source$ ls
netanim-3.108  ns-3.31  pybindgen  test2.cpp  test3.cpp
eece@ubuntu:~/source$ g++ test3.cpp
eece@ubuntu:~/source$ ./a.out
In the No.1 time, the randomized quicksort spends 0.018ms running.
In the No.2 time, the randomized quicksort spends 0.017ms running.
In the No.3 time, the randomized quicksort spends 0.049ms running.
In the No.4 time, the randomized quicksort spends 0.035ms running.
In the No.5 time, the randomized quicksort spends 0.03ms running.
eece@ubuntu:~/source$

```

Question 2

A: Here is the codes of the question:

```
#include<iostream>
using namespace std;

void max_heapify(int A[], int i,int n)
{
    int d, left, right;
    int largest;
    left = 2 * i;
    right = 2 * i + 1;
    if ((left<=n) && (A[left] > A[i]))
        largest = left;
    else
        largest = i;
    if ((right<=n)&&(A[right]>A[largest]))
        largest = right;
    if (largest != i)
    {
        d = A[i];
        A[i] = A[largest];
        A[largest] = d;
        max_heapify(A, largest, n);
    }
}

void build_max_heap(int A[],int n)
{
    int i;
    for (i = n / 2; i >= 1; i--)
    {
        max_heapify(A,i,n);
    }
}

void print_vector(int v[], int n)
{
    int i;
    cout << "Vector:";
    for (i = 1; i <=n; i++)
        cout << " " << v[i];
    cout << endl;
```

```

}

int main()
{
    int A[101];
    int i,d,n;
    int largest;
    for (i = 1; i <=100; i++)
        A[i] = i;
    print_vector(A, 100);
    for (i = 1; i <=100; i++)
    {
        int num = rand() % 100+1;
        d = A[i];
        A[i] = A[num];
        A[num] = d;
    }
    d = 0;
    n = 100;
    print_vector(A, 100);
    build_max_heap(A, 100);
    for (i = 100; i >= 2; i--)
    {
        d = A[1];
        A[1] = A[i];
        A[i] = d;
        n--;
        max_heapify(A, 1, n);
    }
    print_vector(A, 100);
}

```

The result of the codes is the picture below:

```

vector: 4 78 23 41 22 33 77 59 90 65 25 31 36 28 62 92 61 43 97 37 16 27 18 56 93 83 6 96 14 99 48 12 72 44 5 15 2 40 30
21 86 38 20 39 73 10 71 51 80 69 76 74 9 58 84 60 26 47 17 7 19 34 91 35 50 79 87 64 89 24 75 11 53 8 46 1 88 68 94 3 4
5 57 98 100 54 67 42 66 70 63 52 95 32 13 55 82 29 85 81 49
vector: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 4
1 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 8
1 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

```

Question3

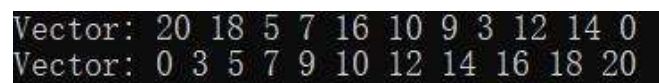
A: Here is the codes of the question:

```
#include<iostream>
using namespace std;

void print_vector(int v[], int n)
{
    int i;
    cout << "Vector:";
    for (i = 0; i < n; i++)
        cout << " " << v[i];
    cout << endl;
}

int main()
{
    int A[11] = {20,18,5,7,16,10,9,3,12,14,0 };
    int B[11];
    int C[21];
    int i,j;
    print_vector(A, 11);
    for (i = 0; i < 21; i++)
        C[i] = 0;
    for (j = 0; j < 11; j++)
        C[A[j]] = C[A[j]] + 1;
    for (i = 1; i < 21; i++)
        C[i] = C[i] + C[i - 1];
    for (j = 10; j >= 0; j--)
    {
        B[C[A[j]]-1] = A[j];
        C[A[j]] = C[A[j]] - 1;
    }
    print_vector(B, 11);
}
```

The result of the codes is shown in the picture below:



```
Vector: 20 18 5 7 16 10 9 3 12 14 0
Vector: 0 3 5 7 9 10 12 14 16 18 20
```

Question4:

A: Here is the codes of the question:

```
#include<iostream>
using namespace std;

void print_vector(int v[], int n)
{
    int i;
    cout << "Vector:";
    for (i = 0; i < n; i++)
        cout << " " << v[i];
    cout << endl;
}

int main()
{
    int A[7] = { 329,457,657,839,436,720,353 };
    int B[7];
    int C[10];
    int i, j;
    print_vector(A, 7);
    for(i=0;i<10;i++)
        C[i] = 0;
    for (j = 0; j < 7; j++)
        C[(A[j] % 10)] = C[(A[j] % 10)] + 1;
    for (i = 1; i < 10; i++)
        C[i] = C[i] + C[i - 1];
    for (j = 6; j >= 0; j--)
    {
        B[C[(A[j] % 10)] - 1] = A[j];
        C[(A[j] % 10)] = C[(A[j] % 10)] - 1;
    }
    for (j = 0; j < 7; j++)
    {
        A[j] = B[j];
    }
    for (i = 0; i < 10; i++)
        C[i] = 0;
    for (j = 0; j < 7; j++)
        C[((A[j]/10)%10)] = C[((A[j] / 10) % 10)] + 1;
    for (i = 1; i < 10; i++)
        C[i] = C[i] + C[i - 1];
}
```

```

for (j = 6; j >= 0; j--)
{
    B[C[((A[j] / 10) % 10) - 1] = A[j];
    C[((A[j] / 10) % 10)] = C[((A[j] / 10) % 10)] - 1;
}
for (j = 0; j < 7; j++)
{
    A[j] = B[j];
}
for (i = 0; i < 10; i++)
    C[i] = 0;
for (j = 0; j < 7; j++)
    C[(A[j]/100)] = C[(A[j] / 100)] + 1;
for (i = 1; i < 10; i++)
    C[i] = C[i] + C[i - 1];
for (j = 6; j >= 0; j--)
{
    B[C[(A[j] / 100)] - 1] = A[j];
    C[(A[j] / 100)] = C[(A[j] / 100)] - 1;
}

print_vector(B, 7);
}

```

The result of the codes is shown in the picture below:

```

Vector: 329 457 657 839 436 720 353
Vector: 329 353 436 457 657 720 839

```