Concepts of Algorithms
CS 211

Exercises & **Solutions**

Semester: 471

Qassim University جامعة القصيم
College of Computer كلية الحاسب

Exercise 1

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{   k = k + n / 2; }
```

Total cost=

Time complexity:

Solution:

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{ k = k + n / 2; }
```

C1 1
C2 n/2+2
C3 n/2+1

Total cost= C1+C2*(n/2+2)+C3 *(n/2+1)
= 1/2(C2+C3)n+(C1+2C2+C3)
=a*n+b
Time complexity= O(n)

Exercise 2

```
int a = 0;
i = M;

while (i > 1) {

    a += i;

    i /= 2;}
```

Total cost=

Time complexity:

Solution:

```
int a = 0;
i = M;

while (i > 1) {

    a += i;

    i /= 2;

}
```

C1 1
C2 1
C3 $logm + 1$
C4 logm
C5 logm

Time complexity: O(logm)

Exercise 3

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{
for (j = 2; j <= n; j = j * 2)

{ k = k + n / 2; }
 }
```

Total cost=

Time complexity:

Solution:

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{
for (j = 2; j <= n; j = j * 2)

{ k = k + n / 2; }
 }
```

C1 1
C2 n/2 +2
C3 logn+1 *n/2 +1

C4 logn*n/2 +1

Total cost= C1+C2*n/2+2+C3*(logn+1 *n/2 +1)+C4*(logn*n/2 +1)
= C1+C2*n/2+ 2C2+C3*(logn*n/2 + logn+ n/2 +1)+C4*(logn*n/2 + logn)
= C1+C2*n/2+ 2C2+C3 logn*n/2 + C3 logn+ C3 n/2 + C3+C4logn*n/2 + C4 logn)
=(C4+C3) logn+1/2(C3+C2)n+1/2(C3+C4)nlogn+(C1+ 2C2+ C3)
=a*logn+b*n+c*nlogn+d

Time complexity: O(nlogn)

Exercise 4

```
int value = 0;
for(int i=0;i<n;i++)
    for(int j=0;j<i;j++)
      value += 1;
```

Total cost=

Time complexity:

Solution:

```
int value = 0;
for(int i=0;i<n;i++)
    for(int j=0;j<i;j++)
      value += 1;
```

C1 1

C2 n+1

C3 $\sum_{j=0}^{n-1} j + 1$

C4 $\sum_{j=0}^{n-1} j$

Total cost= c1+c2*(n+1)+c3*($\sum_{j=0}^{n-1} j + 1$)+c4*($\sum_{j=0}^{n-1} j$)

Time complexity: $O(n^2)$

Exercise 5

```
for (int i = 0; i < m; i = i + 1)
{  x=x+1,
}
for(int i = 0; i < n; i = i + 1)
 {k=k+1,  }
```

Total cost=

Time complexity:

Solution:

```
for (int i = 0; i < m; i = i + 1)
{  x=x+1,
}
for(int i = 0; i < n; i = i + 1)
 {k=k+1,  }
```

C1 m+1
C2 m
C3 $n + 1$
C4 n

Total cost= C1*(m+1)+C2*m+C3*(n+1)+C4*n
=C1*+C1+C2*m+C3*n+ C3+C4*n
= C2*m+C3*n+(C1+C3)
=a*m+b*n+c

Time complexity: O(m+n)

Exercise 6

```
count = 0;
for (int i = N; i > 1; i /= 2)
{count++;}
```

Total cost=

Time complexity:

Solution:

```
count = 0;
for (int i = N; i > 1; i /= 2)
{count++;}
```

C1	1
C2	logn+1
C3	$logn$

Total cost= C1+C2logn+C2+C3logn
= (C2+C3)logn+(C1+C2)
=a*logn+b

Time complexity: O(logn)

Exercise 7

- If an algorithm takes 3 seconds to run with the problem size 16, what is the time requirement (approximately) for that algorithm with the problem size 64?

- If its order is:

  O(1) ➔ T(n) = 3 seconds

  O(n*log$_2$n) ➔ T(n) =3*(64*log64)/(16*log16)=18 seconds

  O(n^3) ➔ T(n) =3*(64^3)/(16^3)=192 seconds

Exercise 8

```
int linearSearch(int arr[], int n,
int key) {
    for (int i = 0; i < n; i++) {
        if (arr[i] == key) {
            return i;
        }
    }
    return -1;
}
```

Total cost=

Time complexity:

Solution:

```
int linearSearch(int arr[], int n,
int key) {
    for (int i = 0; i < n; i++) {
        if (arr[i] == key) {
            return i;
        }
    }
    return -1;
}
```

C1	1
C2	n+1
C3	n
C4	1
C5	1

Total cost= C1+C2*n+C2+C3*n+C4+C5

= (C2+C3)n+(C1+C2+C4+C5)

=a*n+b

Time complexity: O(n)

Exercise 9

```
void printPairs(int arr[], int
n)}
    for (int i = 0; i < n; i++)}
        for (int j = 0; j < n;
j++)}
            printf("(%d, %d)\n",
arr[i], arr[j]);
{
{
}
```

Total cost=

Time complexity:

Solution:

```
void printPairs(int arr[], int
n)}
    for (int i = 0; i < n; i++)}
        for (int j = 0; j < n;
j++)}
            printf("(%d, %d)\n",
arr[i], arr[j]);
{
{
}
```

C1 1
C2 n+1
C3 $n + 1 * n$
C4 n*n

Total cost= C1+C2*n+C2+ C3*n^2 +C3*n+C4*n^2
= (C3+C4)n^2(C2+C3)n+(C1+C2)
=a*n^2+b*n+c

Time complexity: O(n^2)

Exercise 10

```
int findMax(int arr[], int n) {
    int max = arr[0]; // Step 1

    for (int i = 1; i < n; i++) {
// Step 2
        if (arr[i] > max) { //
Step 3
            max = arr[i]; // Step
4
        }
    }

    return max; // Step 5
}
```

Total cost=

Time complexity=

Solution:

```
int findMax(int arr[], int n) {
    int max = arr[0]; // Step 1

    for (int i = 1; i < n; i++) {
// Step 2
        if (arr[i] > max) { //
Step 3
            max = arr[i]; // Step
4
        }
    }

    return max; // Step 5
}
```

C1 1
C2 1
C3 n
C4 n-1
C5 n-1
C6 1

Total cost= C1+C2+C3*n+ C4*n-C4 +C5*n-C5+C6
= (C3+C4+C5)n+(C1+C2-C4 -C5+C6)
=a*n+c
Time complexity= O(n)

Exercise 11

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{ for (j = 2; j <= n; j = j++)
{ k = k + n / 2; }
}
```

Total cost=

Time complexity=

Solution:

```
int i, j, k = 0;
for (i = n / 2; i <= n; i++)
{ for (j = 2; j <= n; j = j++)
{ k = k + n / 2; }
}
```

C1 1
C2 n/2+2
C3. n/2+1∗ n
C4 n/2+1 *n-1

Total cost= C1+C2*(n/2+2)+C3*(n^2/2+n)+ C4 *(n^2/2+n/2-1)
= ½(C3+C4)*n^2 +1/2(C2+C4)+C3*n+(C1+2C2-C4)
=a*n^2+b*n+c
Time complexity= O(n^2)

Exercise 12

```
for (int i = 0; i < N; i++)
 {
    for (int j = 0; j < N; j++)
     { res[i][j] = 0+2;
        for (int z = 0; z < N; z++)
           { res[i][j]+= mat1[i][z]* mat2[z][j];
 }
 }
 }
```

Total cost=

Time complexity=

Solution:

```
for (int i = 0; i < N; i++)
 {
    for (int j = 0; j < N; j++)
     { res[i][j] = 0+2;
        for (int z = 0; z < N; z++)
           { res[i][j]+= mat1[i][z]* mat2[z][j];
 }
 }
 }
```

C1 N+1
C2 N* N+1
C3 N∗ *N*
C4 N*N*(N+1)
C5 N*N*N

Total cost= $C1*(N+1)+C2*(N^2+N)+C3*N^2+C4*(N^3+N^2)+C5*N^3$

$=(C4+C5)*N^3+(C2+C3+C4)*N^2+(C1+C2)*N+C1$

$=a*n^3+b*n^2+c*n+d$

Time complexity= $O(n^3)$

Exercise 13

```
int a= 0;
for (i = 0; i < N; i++)
{ for (j = N; j >i; j--)
{ a = a + i+j; }

}
```

Total cost=

Time complexity:

Solution:

```
int a= 0;
for (i = 0; i < N; i++)
{ for (j = N; j >i; j--)
{ a = a + i+j; }
}
```

C1 1
C2 n+1
C3 $\sum_{j=1}^{n} j + 1$
C4 $\sum_{j=1}^{n} j$ =n(n+1)/2

Total cost= c1+c2*(n+1)+c3*($\sum_{j=1}^{n} j + 1$)+c4*($\sum_{j=1}^{n} j$)

Time complexity: $O(n^2)$

Exercise 14

```
int i =1 ;
 while (i < n)
{  printf("%d ", i);
     i *= 2; }
   printf("\n");
```

Total cost=

Time complexity:

Solution:

```
int i =1 ;
 while (i < n)
{  printf("%d ", i);
     i *= 2; }
   printf("\n");
```

C1 1
C2 log(n)+1
C3 log(n)
C4 log(n)
C5 1

Total cost= C1+C2*(log(n)+1)+C3*log(n) +C4*log(n) +C5
(C2+C3+C4)log(n)+(C1+C2+C5)
a*log(n)+b

Time complexity: O(logn)

Rules for computing times in for-loop

- for (int j = v; j < =vf; j++) → (vf-v+1)+1

```
{
vf-v+1
}
```

- for (int j = v; j < vf; j++) (vf-v)+1

```
{ vf-v

}
```

END