IDMAOA01

– What is the time complexity of the following algorithm with respect to the input size N

```
Algorithm: PolyEvaluate(C,n,x)
Input: An array C stores n real numbers that are the
coefficients of the polynomial. A real value x.
Output: The value of f(x).

S←0
For i ← 0 to n-1 do
{
  P←1
  If (i <> 0)
    For (k ← 1 to i) do
       P←p*x
  S←s+C[i]*p
}
Return s;
```

Select one:

- O(N)
- O(2N)
- O(N^2)
- O(1)

IDMAOA02

– What is the time complexity of the following algorithm with respect to the input size N

```
Algorithm:Search(A,n,m)
Input: An array A stores n sorted integer. An integer m.
Output: An integer i that a[i]=m.
        -1 if m doesn't appear in the array.

For i ← 0 to n-1 do
  If (a[i] = m) then
    Return i;
```

Select one:

- O(N^2)
- O(N-1)

- O(N)
- O(2N)

IDMAOA03

– What is the time complexity of the following algorithm with respect to the input size N

```
Algorithm: GCD(m,n)
Input: Two integers m and n
Output: The value of gcd(m,n)

i←n
While (i>1) do
  If (m is divisible by i) AND (n is divisible by i)
    Return i
  i--
Return 1
```

Select one:

- O(M)
- O(N+M)
- O(N)
- O(N*M)

IDMAOA05

– What is the time complexity of the following algorithm with respect to the input size N

```
Algorithm sum(n)
Input: an integer n
Output: the sum S = ∑_{i=1}^{n} i^3

    s ← 0
    for  i← 1 to n do
         s=s+i*i*i;
    return s;
```

Select one:

- O(1)
- O(N^2)
- O(2N)

O(N)

IDMAOA06

– Estimate the time complexity in Big-Oh notation, with respect to the input size N, for the code below:

```
public void reverse()
{
  SLNode prev=null;
  SLNode current = head;
  SLNode tmp;
  while (current !=null)
  {
    tmp=current.getNext();
    current.setNext(prev);
    prev=current;
    current=tmp;
  }
  __________;
}
```

Select one:

O(N^2)

O(1)

O(N)

O(N^3)

IDMAOA08

– Estimate the time complexity in Big-Oh notation, with respect to the input size N for the code below

```
for (i = 0; i < N; i++)
     for (j = 0; j < N * N; j++)
      sum++;
```

Select one:

O(1)

O(N^2)

- O(N^3)
- O(N)

IDMAOA10

– Estimate the time complexity in Big-Oh notation, with respect to the input size N for the code below

```
for (i = 1; i < N; i++)
   for(j = 1; j < i * i; j++)
       sum++
```

Select one:

- O(N^2)
- O(N)
- O(1)
- O(N^3)

IDMAOA11

– Estimate the time complexity in Big-Oh notation, with respect to the input size N for the code below

```
for (i = 0; i < n; i++)
     for (j = 0; j < i * i; j++)
       for (k = 0; k < j; k++)
          sum++;
```

Select one:

- O(N^4)
- O(N^2)
- O(N^5)
- O(N^3)

IDMAOA12

– Consider two algorithms which have the time complexity in Big-Oh notation below. Which statement is correct?

Select one:

- Two algorithm are equivalent in term of time efficiency.
- The second algorithm is four times faster than the first one.
- With the same value of N, two algorithm have the same computational steps.
- The first algorithm is four times faster than the second one.

IDMAOA13

– Consider three algorithms which have the time complexity in Big-Oh notation below. Please arrange these algorithms in the ascending order of time efficiency (the slowest algorithm is the first one in the order).

$f_1(N)=O(2N^5 + N)$; $f_2(N)=O(N\log N)$; $f_3(N)=O(2^N)$

Select one:

- Algorithm 1, Algorithm 3, Algorithm 2
- Algorithm 1, Algorithm 2, Algorithm 3
- Algorithm 3, Algorithm 2, Algorithm 1
- Algorithm 2, Algorithm 3, Algorithm 1
- Algorithm 2, Algorithm 1, Algorithm 3
- Algorithm 3, Algorithm 1, Algorithm 2

IDMAOA14

– Consider three algorithm which have the time complexity in Big-Oh notation below. Please arrange these algorithms in the descending order of time efficiency (the fastest algorithm is the first one in the order).

$f_1(N)=O(N\log N)$; $f_2(N)=O(\log N)$; $f_3(N)=O(N)$

Select one:

- Algorithm 2, Algorithm 3, Algorithm 1
- Algorithm 1, Algorithm 2, Algorithm 3
- Algorithm 2, Algorithm 1, Algorithm 3
- Algorithm 1, Algorithm 3, Algorithm 2

- Algorithm 3, Algorithm 2, Algorithm 1
- Algorithm 3, Algorithm 1, Algorithm 2

IDMAOA15

– Estimate the time complexity in Big-Oh notation, with respect to the input size N for the code below

```
int x=0;
for (i=0; i<n; i++)
{
      int j=n/2;
      while (j>0)
      {
            x=x*i;
            j--;
      }
}
return x;
```

Select one:

- O(2N)
- O(N)
- O(N^2)
- O(N/2)