

THI_KTHP_HK1_21_22_CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT_31524_7g_14/01/2022

🕒 60 phút 🗞 50 câu hỏi ✔ Điểm qua vòng: Bắt đầu: 07:00 14/01/2022 | Kết thúc: 08:00 14/01/2022

5

THI_KTHP_HK1_21_22_CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT_31524_7g_14/01/2022

019101036 Bùi Thị Thùy Dương

Điểm trắc nghiệm

7.20 / 10.00

Tổng điểm

7.2 / 10

KẾT QUẢ

ĐẠT

Làm bài lúc: 07:00 14/01/2022

Nộp bài lúc: 08:00 14/01/2022

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
41	42	43	44	45	46	47	48
49	50						

Ghi chú

📝 Đáp án của bạn

✔ Câu 1 + 0.2 điểm

Cho đoạn code:

```
//Lấy nội dung từ stack
public int Pop(out bool error)
{
    int result;
    if (IsEmpty()) //stack rỗng
    {
        error = true; //không lấy nội dung ra được
        result = 0; //Trả về 0
    }
    else
    {
```

```

    error = false;           // lấy nội dung ra được
    Item p = Sp;             // p là đỉnh stack
    result = p.value;        // trả về nội dung của phần tử p
    1.....;                // Sp trở phần tử sau p
    p = null;                // huỷ bỏ phần tử p
    Count --;               // giảm số phần tử hiện tại
}
return result;
}

```

Hãy cho biết lệnh nào sau đây được đưa vào dòng lệnh số{1} trong đoạn code trên?

- A. ☐ Sp = p.Value
 - B. ☒ Sp = p.Next
 - C. ☐ p = Sp.Next
 - D. ☐ p = Sp.Value
-

✓ **Câu 2** + 0.2 điểm

Trong giải thuật sắp xếp vun đống, ta có 4 thủ tục con (Insert - thêm 1 phần tử vào cây; Downheap - vun đống lại sau khi loại một phần tử khỏi Heap; Upheap- vun đống sau khi thêm một phần tử vào cây; Remove - loại 1 phần tử khỏi cây nhị phân). Để sắp xếp các phần tử trong dãy theo phương pháp vun đống, ta thực hiện 4 thủ tục trên theo thứ tự như thế nào?

- A. ☐ Insert –Downheap – Remove – Unheap
 - B. ☐ Insert – Remove – Unheap – Downheap
 - C. ☒ Insert – Unheap – Remove – Downheap
 - D. ☐ Remove – Unheap – Insert – Downheap
-

✓ **Câu 3** + 0.2 điểm

Cho phương thức **RemoveOrder**: loại bỏ khoá trong danh sách liên kết đơn có thứ tự tăng dần, hãy cho biết dòng lệnh nào sau đây được thêm vào vị trí số 1 và vị trí số 2 trong phương thức **RemoveOrder**?

```
//error: false nếu loại bỏ được (tìm thấy)
//      : true nếu không loại bỏ được (không tìm thấy)
public void RemoveOrder (int X, out bool error)
{
    //Tìm X trong danh sách
    bool found = false;    //Chưa tìm thấy
    Item tp = null;        //tp là phần tử ngay trước p
    Item p = First;        //p là phần tử đầu tiên
    //p là phần tử của danh sách và chưa tìm thấy
    while ((p != null) && (! found))
    {
        if (p.value < X)    // Khoá của p < value
        {
            tp = p;        // tp đến p
            p = p.Next;    // p đến phần tử kế tiếp
        }
        else if (p.value == X)    //Khoá của p = X
            found = true;        //Tìm thấy
        else
            p = null;            //p đến hết danh sách
    }
    if (found)                //Tìm thấy khoá X
    {
        if (p == First)
        {
            First = p.Next;
            p.Next = null;
        }
        else
        {
            tp.Next = p.Next;
            p.Next = null;
        }
    }
    error = ! found;
}
```

```

//Loại bỏ phần tử p
if ( p == First)
    1. ....;
else
    2. ....;
p = null;
Count --;
}
error = !found;
}

```

// p là phần tử đầu tiên
// First là phần tử sau p
//p không là phần tử đầu tiên
//Phần tử tp trở phần tử sau p
//hủy bỏ phần tử p
//Giảm số phần tử hiện tại

- A. ☐ First.Next = p.Next; tp.Next = p.Next
- B. ☐ First.Next = p; tp = p.Next
- C. ☒ First = p.Next; tp.Next = p.Next
- D. ☐ First = p ; tp = p.Next

✓ **Câu 4** + 0.2 điểm

Cho mảng A gồm các phần tử {21, 2, 14, 13, 36, 17, 37, 51, 41, 50}. Mảng nào sau đây là Heap từ mảng A?

- A. ☐ 50, 41, 21, 13, 36, 17, 14, 2, 51
- B. ☐ 21, 50, 51, 41, 36, 17, 14, 2, 13
- C. ☒ 51, 50, 21, 41, 36, 17, 14, 2, 13
- D. ☐ 13, 50, 21, 41, 36, 17, 14, 2, 51

✗ **Câu 5** - 0 điểm

Cho dãy số [6, 3, 5, 4, 1]. Áp dụng phương pháp sắp xếp chèn tăng dần. Dãy số thu được sau lần lặp thứ ba là:

- A. ☐ [6, 5, 4, 3, 1]
- B. ☐ [3, 4, 5, 6, 1]
- C. ☐ [6, 3, 5, 4, 1] D. ☐ [3, 5, 6, 4, 1]

✓ **Câu 6** + 0.2 điểm

Trong trường hợp cây nhị phân đầy, số lượng tối đa các nút của cây có chiều cao 8 là:

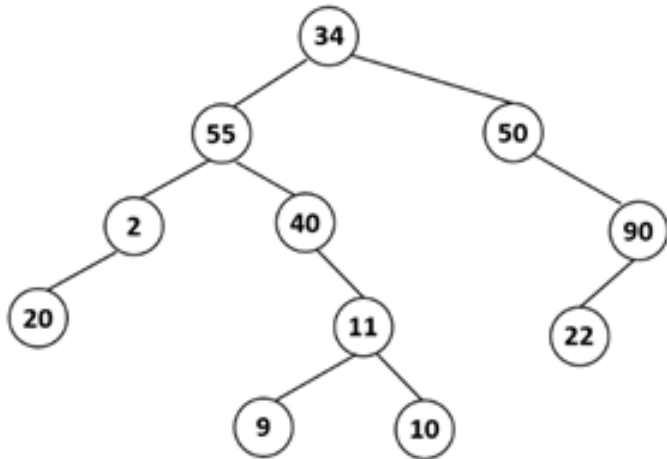
- A. ☐ 255
- B. ☐ 512
- C. ☐ 256

C. ☐ 256

D. ☒ 511

✓ **Câu 7** + 0.2 điểm

Cho cây nhị phân như hình sau. Hãy cho biết kết quả duyệt cây theo các thứ tự duyệt trước (NLR)



A. ☐ 55 2 20 40 11 9 10 50 90 22 34

B. ☒ 34 55 2 20 40 11 9 10 50 90 22

C. ☐ 55 2 20 40 34 11 9 10 50 90 22

D. ☐ 34 55 50 90 22 2 20 40 11 9 10

✗ **Câu 8** - 0 điểm

Nhân tố nào là nhân tố chính ảnh hưởng đến thời gian tính của một giải thuật. Chọn ba đáp án sai bên dưới:

A. ☐ Thuật toán được sử dụng

B. ☒ Chương trình dịch

C. ☒ Kích thước của dữ liệu đầu vào của thuật toán

D. ☐ Cấu trúc dữ liệu

✗ **Câu 9** - 0 điểm

Cho cây nhị phân gồm các nút có khóa là số nguyên. Phương thức **CalTreeHeight** dùng để xác định chiều cao của cây. Các lệnh nào sau đây được đưa vào dòng lệnh thứ 1 và dòng lệnh thứ 2 trong phương thức **CalTreeHeight** sau ?

```
public int CalTreeHeight(Node aRoot) //PostOrder
```

```

{
    if (aRoot == null)
        return -1;
    1 .....;
    int h_right = CalTreeHeight(aRoot.right);
    return
        2.....;
}

```

- A. ☐ `int h_left = CalTreeHeight(aRoot.left); ((h_left < h_right) ? h_left : h_right) + 1`
- B. ☐ `int h_left = CalTreeHeight(aRoot); ((h_left > h_right) ? h_right : h_right) + 1`
- C. ☐ `int h_left = CalTreeHeight(aRoot.left); ((h_left > h_right) ? h_left : h_right) + 1`
- D. ☒ `int h_left = CalTreeHeight(aRoot); ((h_left >= h_right) ? h_left : h_right) + 1`

✖ **Câu 10** - 0 điểm

Từ phương trình xác định thời gian chạy $T(n) = O(f(n))$, hãy xác định cấp ô lớn của $T(n)$ với $T(n) = 3n(n + 6) + 5n + 2^n + 4\log_2 n$

- A. ☐ $O(\log n)$
- B. ☐ $O(2^{\dots})$
- C. ☐ $O(n \log n)$
- D. ☒ $O(2^{\dots \log n})$

✖ **Câu 11** - 0 điểm

Cho dãy số [5, 15, 12, 2, 10, 12, 9, 1, 9, 3, 2, 3]. Áp dụng phương pháp sắp xếp chèn tăng dần. Dãy số thu được sau lần lặp thứ ba là:

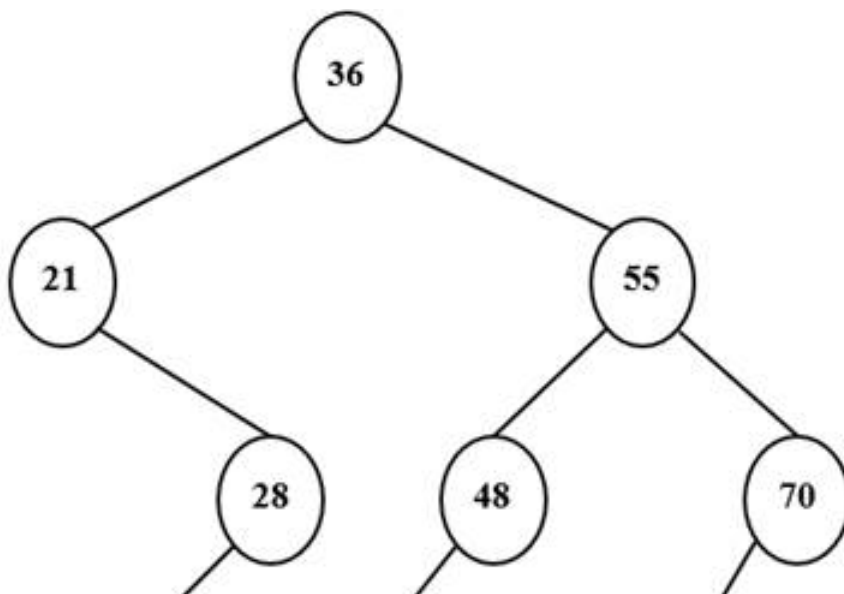
- A. ☐ [2, 5, 12, 15, 10, 3, 2, 3, 12, 9, 1, 9]

B. ☐ [12, 9, 1, 9, 3, 2, 3, 2, 5, 10, 12, 15]

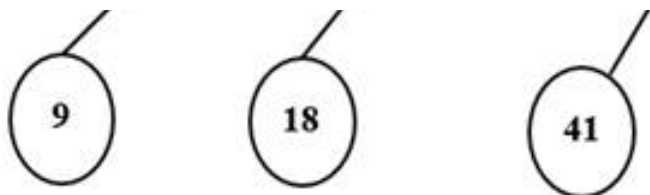
C. ☐ [2, 5, 10, 12, 15, 12, 9, 1, 9, 3, 2, 3] [2, 5, 12, 15, 10, 12, 9, 1, 9, 3, 2, 3]

✖ **Câu 12** - 0 điểm

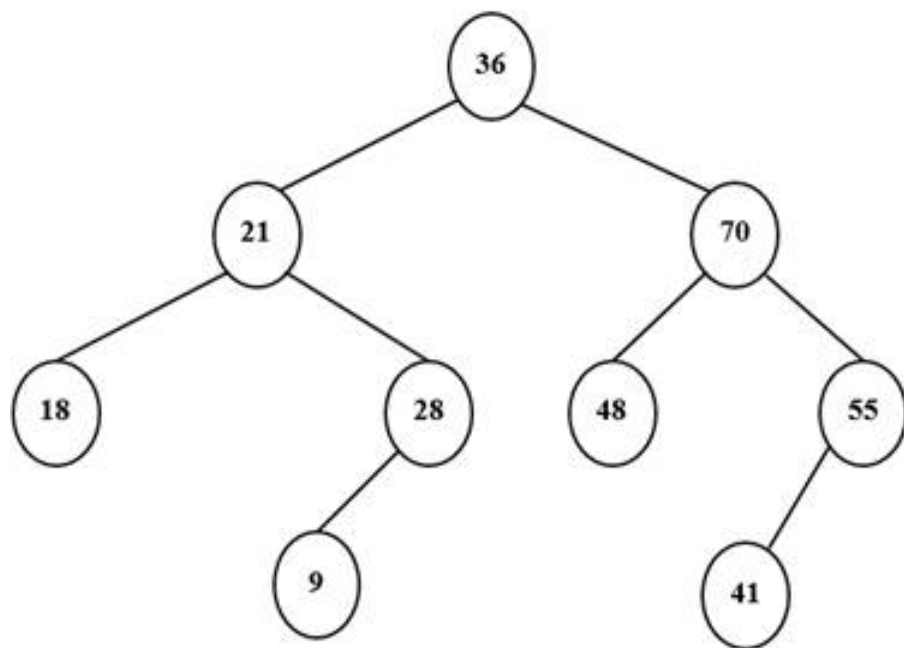
Giả sử cây tìm kiếm nhị phân ban đầu là rỗng, lần lượt chèn các khóa 36, 55, 70, 48, 41, 21, 28, 9, 18. Hãy cho biết cây tìm kiếm nhị phân nào sau đây có được khi chèn vào các khoá trên?



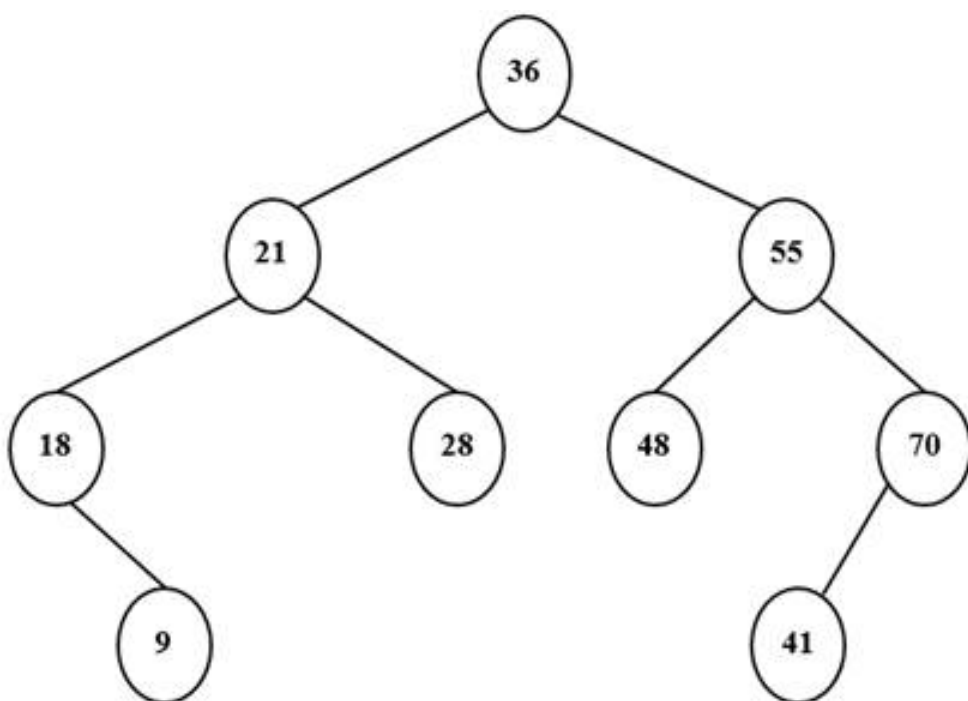
A.



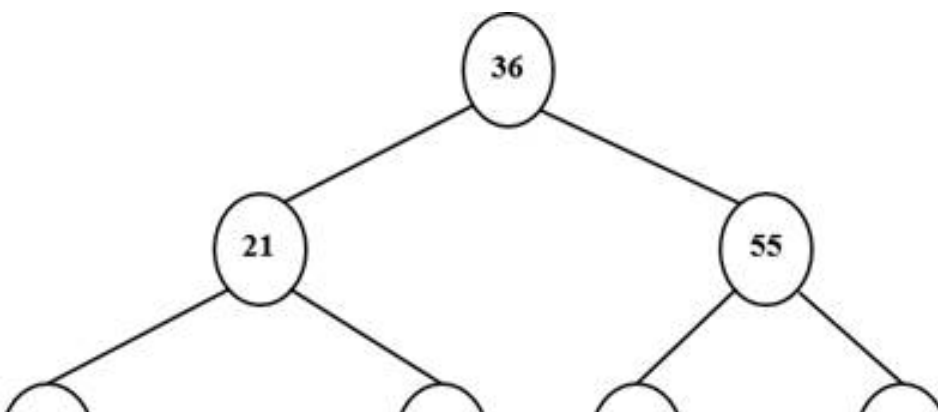
B. ☐

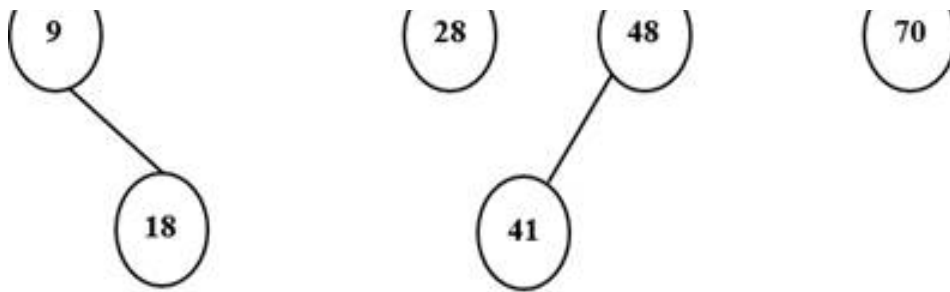


c. ☐



D. ☐





✓ **Câu 13** + 0.2 điểm

Cho dãy số sau: [40, 25, 75, 15, 65, 55, 90, 30, 95, 85]. Áp dụng phương pháp sắp xếp lựa chọn. Dãy số thu được sau lần lặp thứ tư là:

- A. ☐ [40, 65, 55, 90, 15, 25, 75, 30, 95, 85]
- B. ☐ [15, 25, 75, 40, 65, 55, 90, 30, 95, 85]
- C. ☒ [15, 25, 30, 40, 65, 55, 90, 75, 95, 85]
- D. ☐ [40, 25, 75, 15, 25, 75, 30, 95, 85, 95]

✗ **Câu 14** - 0 điểm

Đoạn code dưới đây mô tả thuật toán nào?

```

public void A()
{
    //Xét dãy Arr[i]...Arr[N-1]
    int i = 0;
    bool flag;
    do
    {
        flag = false;
        //Duyệt ngược dãy Arr[i]...Arr[N-1]
        for (int j = N-1; j > i; j--)
            if (Arr[j].key < Arr[j-1].key)
            {
                //Khoá của Arr[j] < khoá của Arr[j-1];
                //Hoàn đổi Arr[j] với Arr[j-1]
                Swap (ref Arr[j], ref Arr[j-1]);
                flag = true;
            }
        i++;
        //Dãy A[0]... A[i-1] có thứ tự
    }
}
  
```

```
}while(flag);  
}
```

- A. ☒ Selection Sort
B. ☐ Insertion Sort
C. ☐ Bubble Sort
D. ☐ Quick Sort
-

✖ **Câu 15** - 0 điểm

Cho dãy số sau: 10, 11, 14, 32, 36, 43, 55, 57, 87, 97 . Áp dụng phương pháp tìm kiếm nhị phân, sau bao nhiêu lần phân đoạn ta sẽ tìm thấy số 43?

- A. ☐ 5
B. ☒ 4
C. ☐ 3
D. ☐ 6
-

✔ **Câu 16** + 0.2 điểm

Giả sử chương trình A có thời gian chạy chương trình $T1(n)=n^2$ (n mũ 2). Giả sử chương trình B có thời gian chạy chương trình $4*n+1$ với n càng lớn thì chương trình tốt hơn chương trình

✔ **Câu 17** + 0.2 điểm

Trong hàng đợi, phép toán Dequeue() dùng để làm gì?

- A. ☐ Xóa phần tử ra khỏi hàng đợi tại giữa hàng đợi
B. ☐ Xóa phần tử ra khỏi hàng đợi tại cuối hàng đợi
C. ☒ Xóa phần tử ra khỏi hàng đợi tại đầu hàng đợi
D. ☐ Thêm phần tử x vào hàng đợi tại cuối hàng đợi
-

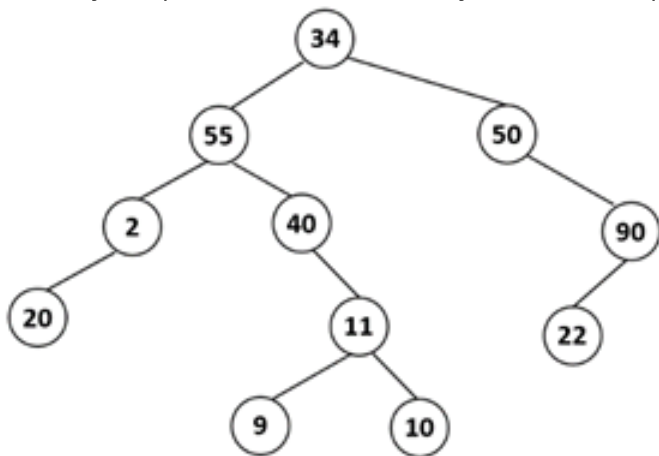
▼ **Câu 18** (+ 0.2 điểm)

Cho dãy số [5, 9, 6, 3, 1]. Áp dụng phương pháp sắp xếp chọn tăng dần. Dãy số thu được sau lần lặp thứ bốn là:


- A. ☐ [5, 1, 9, 6, 3]
B. ☐ [3, 5, 1, 9, 6]
C. ☒ [1, 3, 5, 6, 9]
D. ☐ [5, 1, 3, 6, 9]
-

✓ **Câu 19** (+ 0.2 điểm)

Cho cây nhị phân như hình sau. Hãy cho biết kết quả duyệt cây theo các thứ tự duyệt sau (LRN)




- A. ☐ 34 50 90 22 55 2 20 40 11 9 10
B. ☐ 34 55 2 20 40 11 9 10 50 90 22
C. ☐ 20 2 55 40 9 11 10 34 50 22 90

D.  20 2 9 10 11 40 55 22 90 50 34


✓ **Câu 20** + 0.2 điểm

Hàng đợi là một kiểu dữ liệu trừu tượng (ADT, Abstract Data Type) hoạt động theo cơ chế nào sau đây?

- A.  FIFO (First In First Out)
 - B. ☐ LIFO (Last In First Out)
 - C. ☐ FILO (First In Last Out)
 - D. ☐ LILO (Last In Last Out)
-

✗ **Câu 21** - 0 điểm

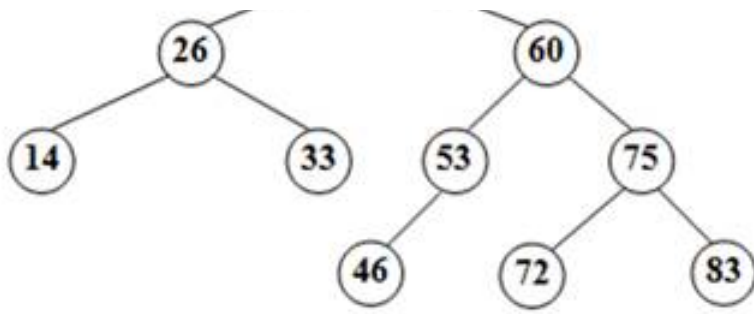
Cho dãy số [4, 0, 2, 8, 5, 9, 6, 1, 3, 7]. Áp dụng phương pháp sắp xếp chèn tăng dần. Dãy số thu được sau lần lặp thứ bốn là:

- A.  [0, 1, 2, 3, 5, 9, 6, 4, 8, 7]
 - B. ☐ [0, 4, 2, 8, 5, 9, 6, 1, 3, 7]
 - C. ☐ [0, 1, 2, 8, 5, 9, 6, 4, 3, 7]
 - D. ☐ [0, 2, 4, 5, 8, 9, 6, 1, 3, 7]
-

✓ **Câu 22** + 0.2 điểm

Cho cây tìm kiếm nhị phân như hình sau. Hãy cho biết kết quả duyệt cây theo các thứ tự sau (LRN)?





- A. ☒ 14 33 26 46 53 72 83 75 60 41
- B. ☐ 41 26 14 33 60 53 46 75 72 83
- C. ☐ 14 26 33 41 46 53 60 72 75 83
- D. ☐ 41 26 60 14 33 53 75 46 72 83
-

✓ **Câu 23** + 0.2 điểm

Để biểu diễn Stack, các kiểu dữ liệu nào sau đây được sử dụng?

- A. ☐ Ngăn xếp
- B. ☒ Danh sách liên kết
- C. ☒ Mảng
- D. ☐ Hàng đợi
-

✓ **Câu 24** + 0.2 điểm

Ghép câu hỏi ở cột 1 với đáp án cột 2

Đáp án của bạn

A. Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 12 thì vị trí của nút con phải là

25

B. Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 12 thì vị trí của nút con trái là

24

C. Cho cây nhị phân T, nhận duyệt cây theo thứ

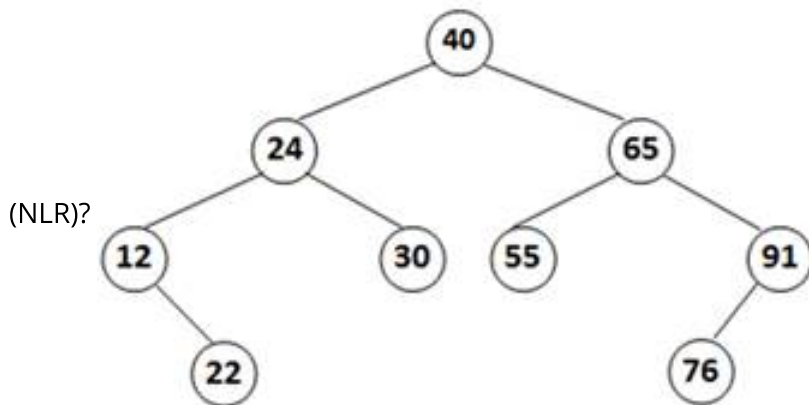
FICGJ

phép duyệt cây theo thứ tự giữa cho kết quả DBHEAFICGJ. Nếu duyệt theo thứ tự sau ta có kết quả : DHEBIFJGCA. Hãy cho biết các nút của cây con phải.

D. Cho cây nhị phân T. BDHE
 Phép duyệt cây theo thứ tự trước cho kết quả ABDEHCFIGJ. Nếu duyệt theo thứ tự giữa ta có kết quả: DBHEAFICGJ. Hãy cho biết các nút của cây con trái:

✓ **Câu 25** + 0.2 điểm

Cho cây tìm kiếm nhị phân như hình sau. Hãy cho biết kết quả duyệt cây theo các thứ tự duyệt trước



- A. ☐ 12 22 24 30 40 55 65 76 91
 B. ☒ 40 24 12 22 30 65 55 91 76
 C. ☐ 22 12 30 24 55 76 91 65 40

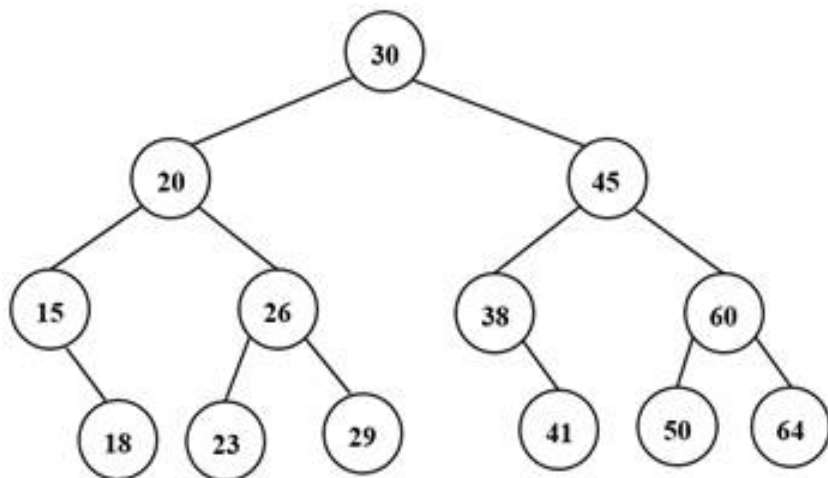
D. ☐ 22 76 12 30 55 91 24 65 40

✓ **Câu 26** + 0.2 điểm

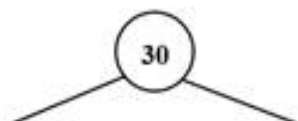
Khi chèn một phần tử vào danh sách (cài đặt mảng) yêu cầu cần độ dài của danh sách phải độ dài của mảng

✓ **Câu 27** + 0.2 điểm

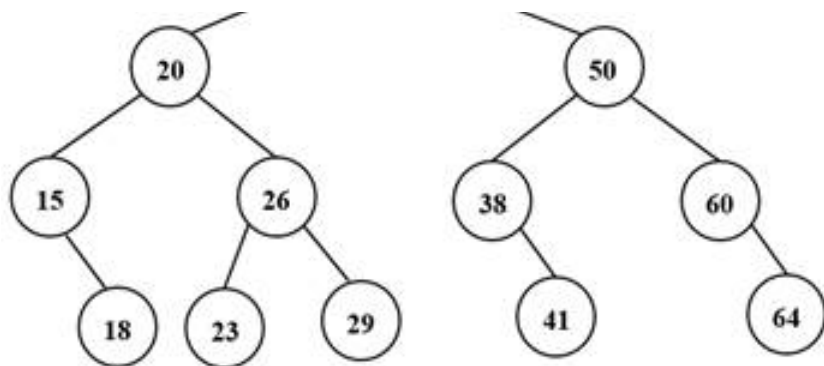
Cho cây tìm kiếm nhị phân như hình sau:



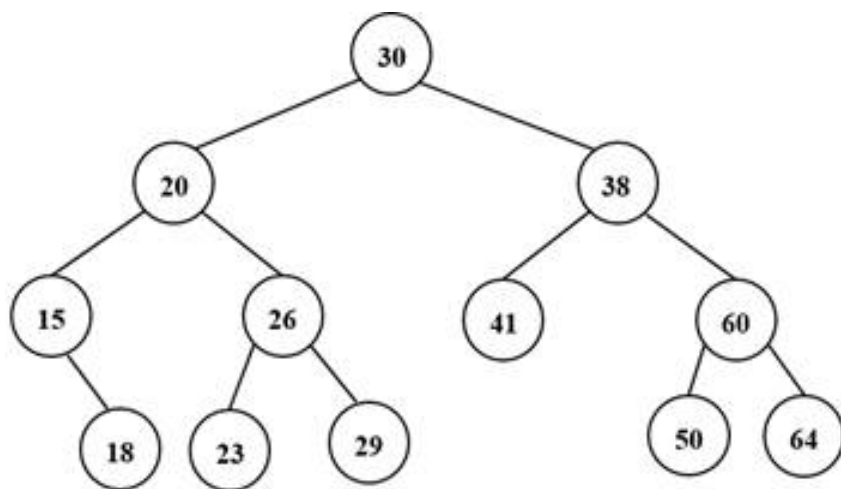
Hình cây nào sau đây khi xóa nút 45?



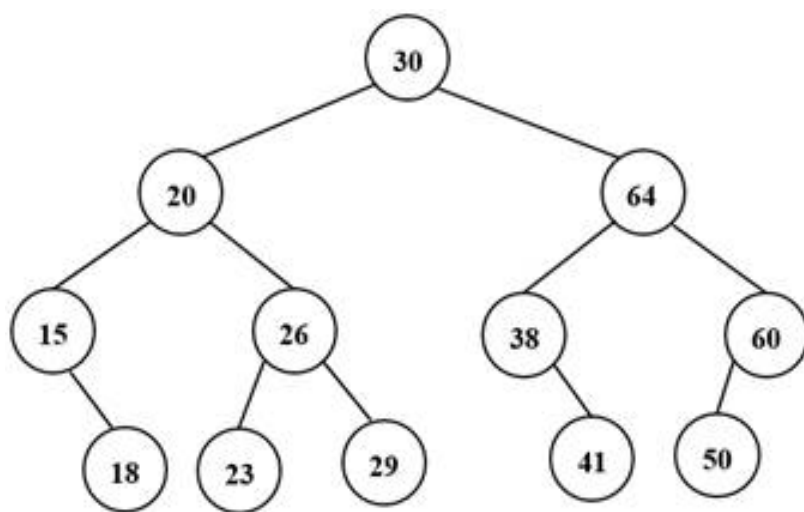
A. 



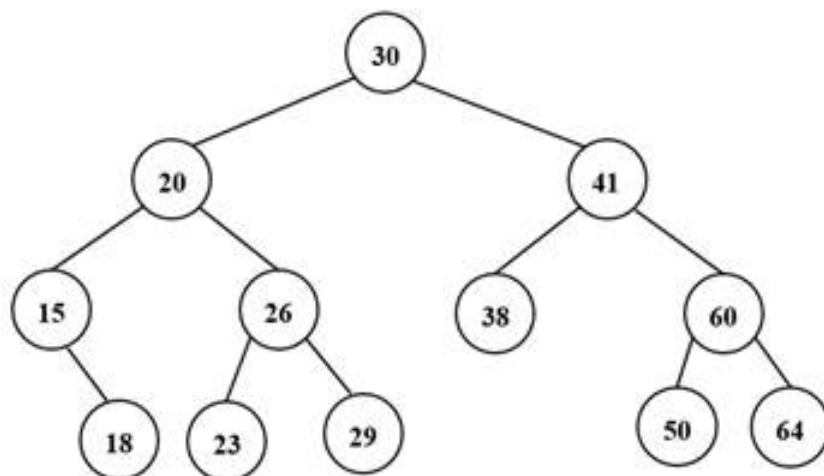
B. ☐



C. ☐



D. 



✓ **Câu 28** + 0.2 điểm

Cho phương thức **InserOrder**: thêm khoá vào đầu danh sách liên kết đơn có thứ tự, hãy cho biết dòng lệnh nào sau đây được thêm vào vị trí số 1 và vị trí số 2 trong phương thức **InserOrder**?

```
public void InserOrder (int X)
{
    //Tìm vị trí thêm X vào danh sách
    bool cont = true;           //Còn tiếp tục
    Item tp = Header;          //tp là phần tử đầu ngay trước p
    Item p = tp.Next;          //p là phần tử đầu tiên
    //p là phần tử của danh sách và còn tiếp tục
    while ((p != null) && cont)
        if (p.value < X)       //Khoá của p < X
        {
            1. ....;           //tp đến p
            2. ....;           //p đến phần tử kế tiếp
        }
        else
        {
            cont = false;       //Không tiếp tục
            //Tạo phần tử mới q có khoá là X
            Item q = new Item (X);
            //Thêm phần tử q vào giữa tp và p
            tp.Next = q;         //Phần tử tp trỏ q
            q.Next = p;          //Phần tử q trỏ p
            Count ++;           //Tăng số phần tử hiện tại
        }
}
```

- A. ☐ tp = p.Next; p.Next = p
- B. ☐ tp.Next = p; p = p.Next
- C. ☒ tp = p; p = p.Next
- D. ☐ tp.Next = p.Next; p = p.Next

✗ **Câu 29** - 0 điểm

Giả sử có file A chứa các số nguyên gồm {22, 5, 17, 9, 35, 26, 43, 25, 32, 7, 15}. Quá trình sắp xếp các số trên file A bằng trộn 2 đường trực tiếp với 2 file tạm là file B và file C. Hãy cho biết run của mảng A sau khi trộn từ file B và file C với len = 2?

- A. ☐ 5, 9, 17, 22, 25, 26, 35, 43, 7, 15, 32
- B. ☐ 5, 9, 17, 22, 25, 26, 35, 43, 7, 15, 32
- C. ☒ 5, 7, 9, 15, 17, 22, 25, 26, 32, 35, 43
- D. ☐ 5, 22, 9, 17, 26, 35, 25, 43, 7, 32, 15

✓ **Câu 30** + 0.2 điểm

```
public void A ()
{
    int start, end;
    Stack stack = new Stack();
    // Dãy Arr[0]...Arr[N-1];
    stack.Push(0, N-1);
    while (!stack.IsEmpty())
    {
        stack.Pop(out start, out end);
        while (start < end)
        {
            Item pivot = Arr[(start + end )/2].key;
            int i = start;
            int j = end;
            while (i < j)
            {
                while ( Arr[i].key < pivot)
                    i++;
                while (Arr[j].key > pivot)
                    j--;
                if ( i <= j)
                {
                    Swap (ref Arr[i], ref Arr[j]);
                }
            }
        }
    }
}
```

Đoạn code dưới đây mô tả thuật toán nào?

```

        i++;
        j--;
    }
}
if (i < end)
    //Dãy Arr[i]...Arr[end] có nhiều hơn 1 phần tử
    stack.Push(i, end);
    //Phân đoạn dãy Arr[start]... Arr[j]
    end = j;
}
}
}

```

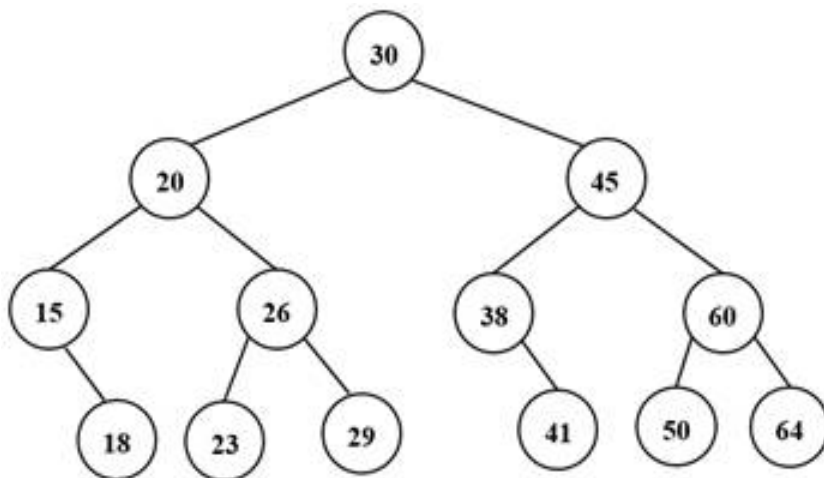
- A. ☐ Selection Sort
- B. ☐ Insertion Sort
- C. ☐ Bubble Sort
- D. ☒ Quick Sort

✓ Câu 31 + 0.2 điểm

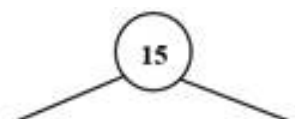
Dùng STACK để lưu trữ số nhị phân có giá trị bằng số thập phân 215 ta có kết quả

✓ Câu 32 + 0.2 điểm

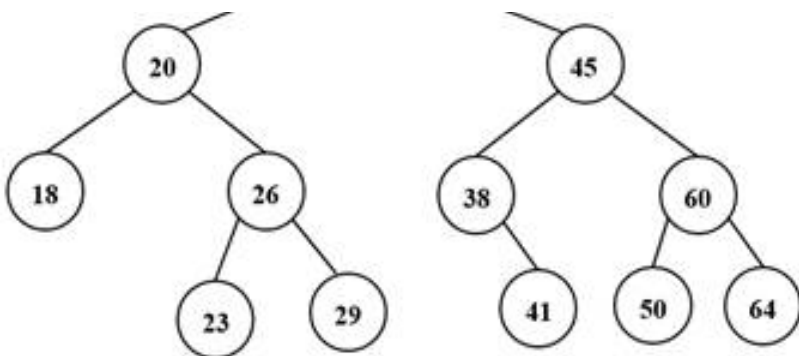
Cho cây tìm kiếm nhị phân như hình sau:



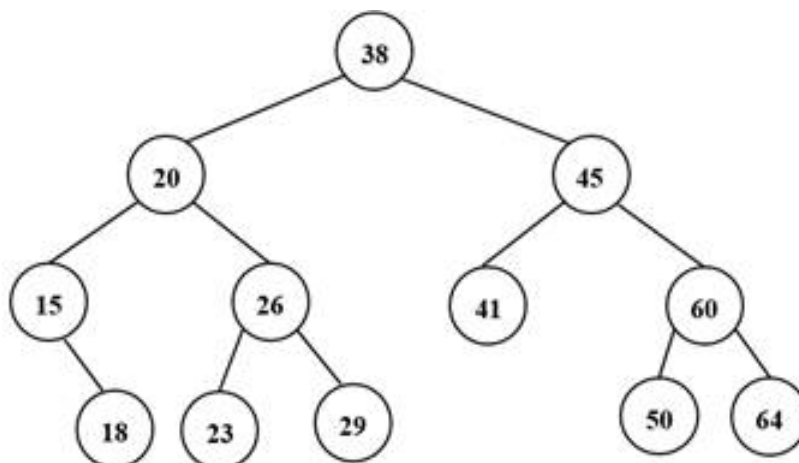
Hình cây nào sau đây khi xóa nút 30?



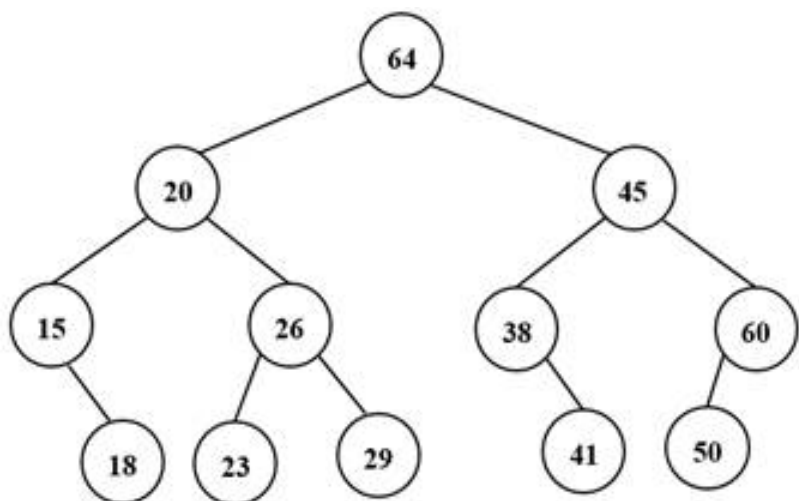
A. ☐



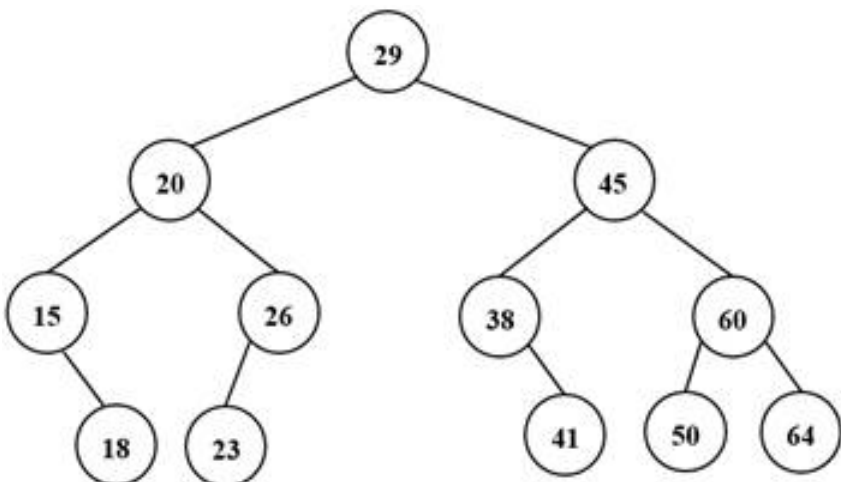
B. ☒



C. ☐



D. ☒



Cho dãy số [4, 0, 2, 8, 5, 9, 6, 1, 3, 7]. Áp dụng phương pháp sắp xếp chèn tăng dần. Dãy số thu được sau lần lặp thứ ba là:

- A. ☐ [0, 2, 3, 8, 5, 9, 6, 1, 4, 7]
 - B. ☐ [0, 2, 4, 5, 8, 9, 6, 1, 3, 7]
 - C. ☐ [0, 1, 2, 8, 5, 9, 6, 4, 3, 7]
 - D. ☒ [0, 2, 4, 8, 5, 9, 6, 1, 3, 7]
-

✓ Câu 34 + 0.2 điểm

Cho dãy số [5, 15, 12, 2, 10, 12, 9, 1, 9, 3, 2, 3]. Áp dụng phương pháp sắp xếp chèn tăng dần. Dãy số thu được sau lần lặp thứ năm là:

- A. ☒ [2, 5, 10, 12, 12, 15, 9, 1, 9, 3, 2, 3]
 - B. ☐ [2, 5, 12, 15, 10, 12, 9, 1, 9, 3, 2, 3]
 - C. ☐ [2, 5, 12, 15, 10, 3, 2, 3, 12, 9, 1, 9]
 - D. ☐ [12, 9, 1, 9, 3, 2, 3, 2, 5, 10, 12, 15]
-

✓ Câu 35 + 0.2 điểm

Khi cấu trúc dữ liệu thay đổi thì giải thuật cũng thay đổi.

- A. ☒ Đúng
 - B. ☐ Sai
-

✓ Câu 36 + 0.2 điểm

Thuật toán nào sau đây không dựa trên phép so sánh từng cặp phần tử của dãy để quyết định đổi chỗ mà sắp xếp dựa trên sự phân nhóm?

- A. ☐ Merge sort
- B. ☐ Heap sort
- C. ☐ Selection sort
- D. ☒ Radix sort

✖ **Câu 37** - 0 điểm

Ghép câu hỏi ở cột 1 với đáp án cột 2

Đáp án của bạn

A. Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 6 thì vị trí của nút con trái là 12

B. Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 6 thì vị trí của nút con phải là 13

C. Giá trị biểu thức tiền tố: 1
/ * 2 1 + 2 1 * 7 6

1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
5 là

D. Giá trị biểu thức hậu tố -1

1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
/ là

✓ **Câu 38** + 0.2 điểm

Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 3 thì vị trí tương ứng của nút con sẽ là:

- A. ☒ 6
- B. ☒ 7
- C. ☐ 4
- D. ☐ 2

✓ **Câu 39** + 0.2 điểm

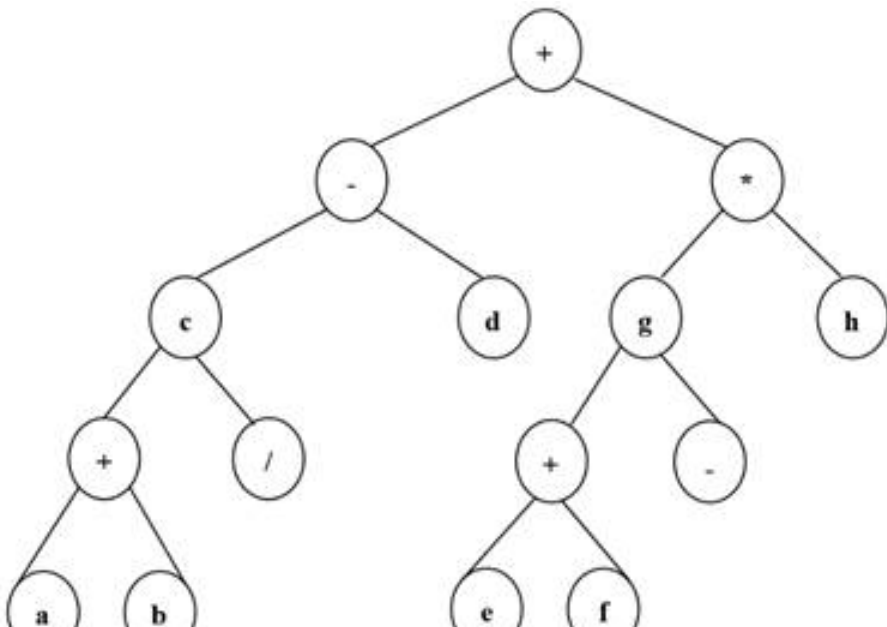
Khi lưu trữ cây nhị phân dưới dạng mảng, nếu vị trí của nút cha trong mảng là 9 thì vị trí của nút con phải là:

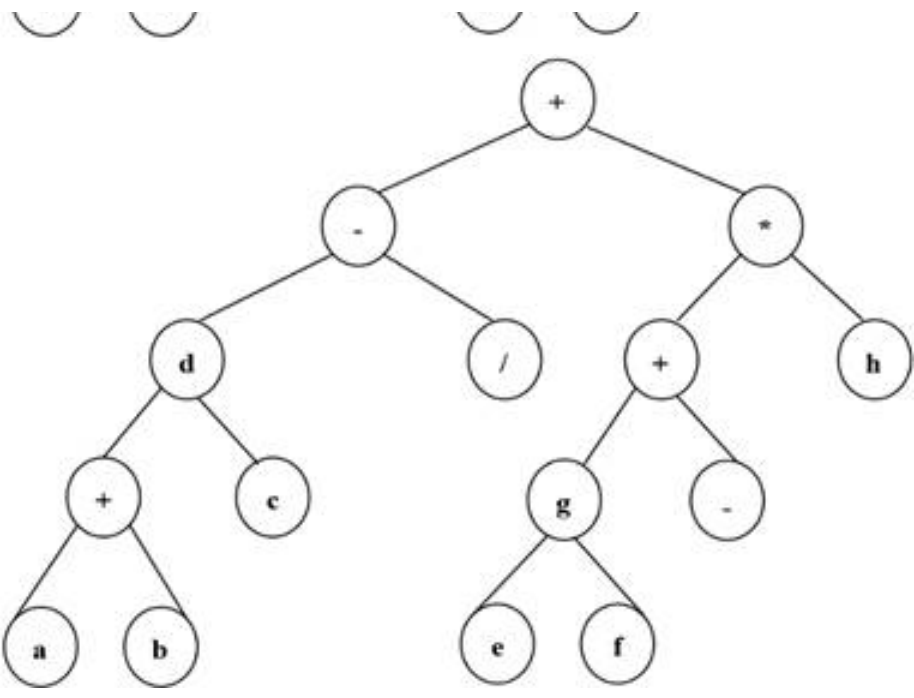
- A. ☒ 19
- B. ☐ 17
- C. ☐ 18
- D. ☐ 10

✓ Câu 40 + 0.2 điểm

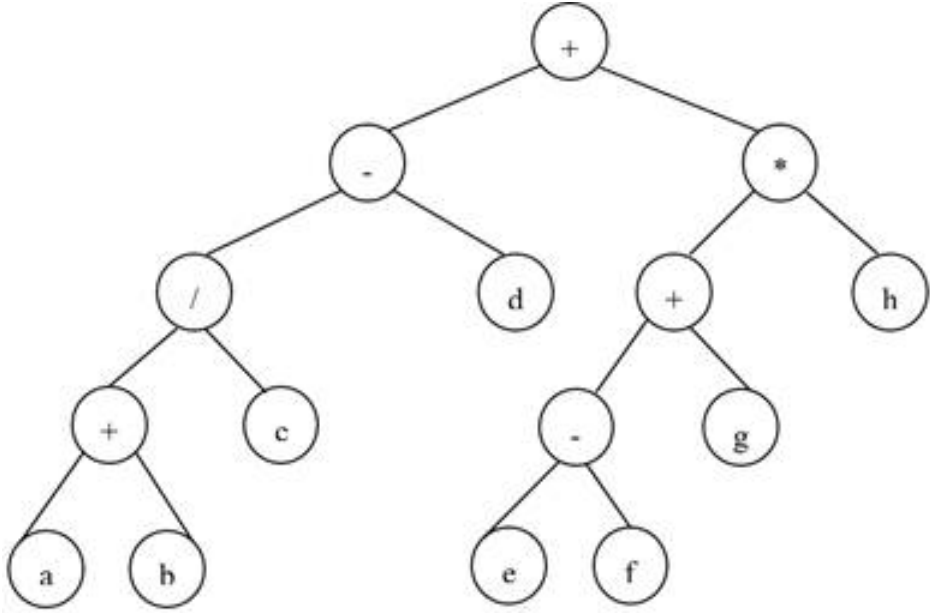
Cho biết hình cây biểu thức biểu diễn biểu thức số học sau: $(a+b) / c - d + (e - f + g) * h$

A. ☐

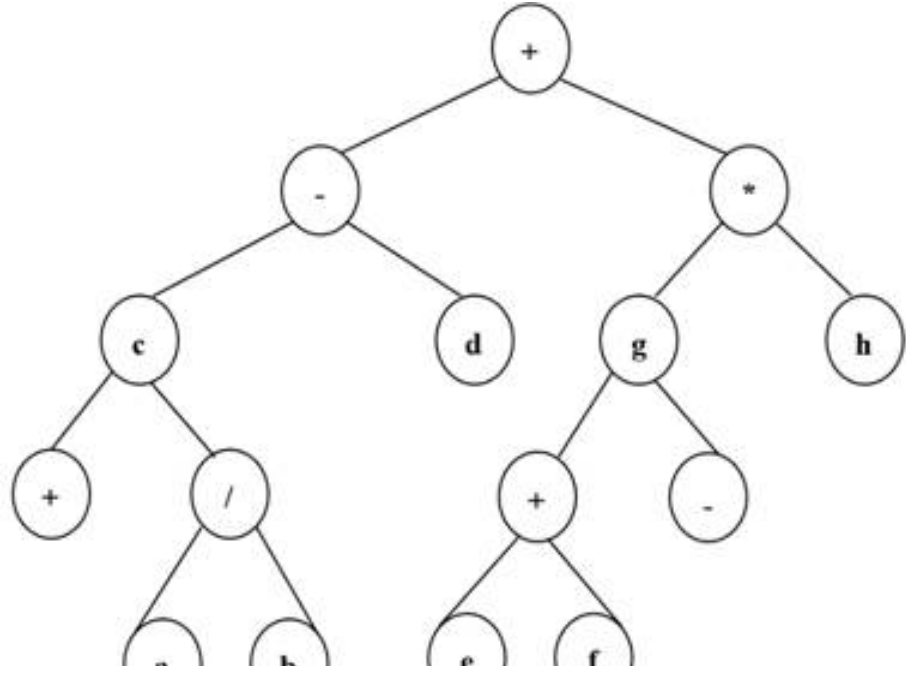




B. ☐



C. ☒



D. ☐

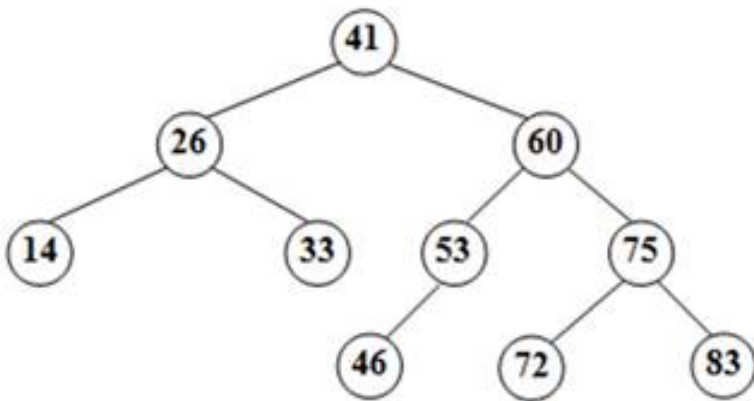


✖ **Câu 41** - 0 điểm

Trong lưu trữ dữ liệu kiểu Queue (Q), giả sử F là con trỏ trỏ tới lối trước của Q, R là con trỏ trỏ tới lối sau của Q. Khi thêm một phần tử vào Queue, thì F không và R =

✔ **Câu 42** + 0.2 điểm

Cho cây tìm kiếm nhị phân như hình sau. Hãy cho biết kết quả duyệt cây theo các thứ tự trước (NLR)?



- A. ☐ 46 72 83 14 33 53 75 26 60 41
- B. ☐ 41 26 60 14 33 53 75 46 72 83
- C. ☐ 14 26 33 41 46 53 60 72 75 83

Key	20		22	33	14	25	15	35	45	18
-----	----	--	----	----	----	----	----	----	----	----

Địa chỉ	0	1	2	3	4	5	6	7	8	9
Key	20	25	22	33	14	45	15	35	28	

D. ☐

✓ **Câu 46** (+ 0.2 điểm)

Khi nói tới dữ liệu thì cần phải xem xét dữ liệu đó cần được thực hiện bằng giải thuật gì để đạt được kết quả như mong muốn.

- A. ☒ Đúng
- B. ☐ Sai

✓ **Câu 47** (+ 0.2 điểm)

Từ phương trình xác định thời gian chạy $T(n) = O(f(n))$, hãy xác định cấp ô lớn của $T(n)$ với $T(n) = n \log(n!) + (3n^2 + 2n) \log n$.

- A. ☐ $O(n \log n)$
- B. ☒ $O(n^2 \log n)$
- C. ☐ $O(n)$
- D. ☐ $O(n^2)$

✗ **Câu 48** (- 0 điểm)

Kiểu dữ liệu trừu tượng là kiểu dữ liệu mà người lập trình phải tự xây dựng dựa trên các kiểu dữ liệu không cơ bản được cung cấp từ ngôn ngữ lập trình.

- A. ☒ Đúng

B. ☐ Sai

✖ **Câu 49** - 0 điểm

Hãy xác định cấp độ lớn cho đoạn mã chương trình sau:

```
1 long factorial(int n)
2 {
3     if (n <= 1)
4         return 1;
5     else
6         return n * factorial(n - 1);
7 }
```

- A. ☐ $O(n)$
B. ☐ $O(\log^2 n)$
C. ☐ $O(n^{\frac{1}{2}})$
D. ☒ $O(n \log n)$
-

✔ **Câu 50** + 0.2 điểm

Để lựa chọn một giải thuật tốt, ta sẽ căn cứ vào các tiêu chuẩn:

- A. ☐ Giải thuật được thực hiện trong một máy tính lý tưởng
B. ☒ Giải thuật đúng đắn
C. ☒ Giải thuật đơn giản
D. ☒ Giải thuật thực hiện nhanh
-