

Berdasarkan tabel operasi *stack* berikut, isi *stack contents* pada baris ke-4 adalah ....

No	Method	Return Value	Stack Contents
1	push (7)	-	
2	push (9)	-	
3	top ()	9	
4	push (4)	-	

- A. (4)
- B. (7, 9)
- C. (7, 9, 4)
- D. (4, 7, 9)

Berdasarkan ilustrasi rangkaian *linked list* berikut, yang dimaksud dengan *node tail* adalah ....



- A. LAX
- B. MSP
- C. ATL
- D. BOS

Sebuah *node* yang berada di level 0 pada sebuah *Tree* disebut dengan ....

- A. *child*
- B. *subtree*
- C. *root*
- D. *path*

Representasi dari ekspresi aritmatika  $((5 / 3) + 7) + ((3 * 6) * 4)$  dalam *array binary tree* yang benar adalah ....

A. 

5	/	3	+	7	+			3	*	6	*	4			
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

B. 

/	+	+	*	*	5	3	7	3	6	4					
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

C. 

5	/	3	+	7	+			3	*	6	*	4			
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

D. 

+	+	*	/	7	*	4	5	3			3	6			
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

Seorang pengemudi taksi *online* sedang melihat arah jalan untuk menuju pemesan *online* dengan menggunakan handphone. Berdasarkan contoh kasus tersebut, data yang sedang diproses adalah ....

- A. pengemudi
- B. pemesan
- C. arah jalan
- D. handphone

Berdasarkan data berikut (3, 2, 5, 3, 1, 0, 4, 4) dengan rentang data yaitu 0 sampai 5. Jika dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka rangkaian data c yang dihasilkan ....

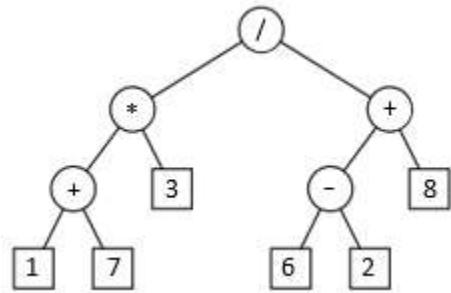
- A. (1, 1, 1, 2, 2, 2)
- B. (0, 1, 2, 3, 4, 5)
- C. (0, 1, 2, 3, 3, 4, 4, 5)
- D. (1, 1, 1, 2, 2, 1)

Berdasarkan tabel operasi *queue* berikut, isi dari *queue contents* baris ke-4 adalah ....

No	Method	Return Value	Queue Contents
1	enqueue(10)	-	
2	enqueue(8)	-	
3	size()	2	
4	dequeue()	10	

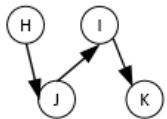
- A. -
- B. 2
- C. 8
- D. 10

Berdasarkan representasiakan *binary tree* berikut, ekspresi aritmatikanya yang tepat adalah ...



- A.  $((1 + 7) * 3) / ((6 - 2) + 8)$
- B.  $((3 * 1) - 7) + ((6 / 2) / 8)$
- C.  $((7 / 1) + 3) - ((2 + 6) * 8)$
- D.  $((1 - 7) * 3) * ((2 * 6) - 8)$

Berdasarkan graph dibawah ini representasi dalam *matriks* yang tepat adalah ....



A. Pilihan A

	H	I	J	K
H	0	1	0	0
I	0	0	0	1
J	0	1	0	0
K	0	0	0	0

B. Pilihan B

	H	I	J	K
H	0	0	1	0
I	0	0	0	1
J	1	0	0	0
K	0	0	0	0

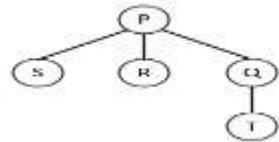
C. Pilihan C

	H	I	J	K
H	0	0	1	0
I	0	0	0	1
J	0	1	0	0
K	0	0	0	0

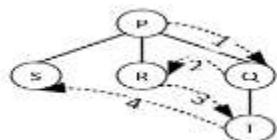
D. Pilihan D

	H	I	J	K
H	0	0	1	0
I	1	0	0	0
J	1	0	0	0
K	0	0	0	0

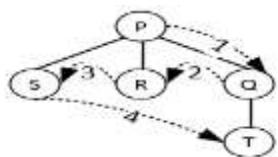
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran pada *graph* berikut adalah ....



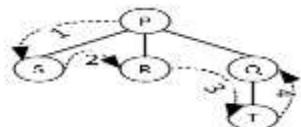
A. Pilihan A



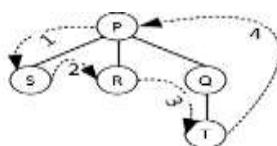
B. Pilihan B



C. Pilihan C



D. Pilihan D



Output dari potongan pada coding BFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(4);
    g.addEdge(1, 0);
    g.addEdge(1, 3);
    g.addEdge(0, 3);
    g.addEdge(3, 1);
    g.addEdge(3, 2);
    g.addEdge(2, 2);
    g.BFS(3);
}
```

- A. 3 2 0 1
- B. 3 1 2 0
- C. 3 0 2 1
- D. 3 2 1 0

Berikut terdapat langkah mengurutkan data dari terkecil ke terbesar dengan menggunakan algoritma *counting sort*.

1. Menghitung banyaknya data yang sama pada rangkaian data awal (rangkaian a)
2. Menyimpan data hasil langkah ke-1 pada rangkain c
3. Menjumlahkan data yang terdapat pada rangkaian c
4. Menyimpan data hasil langkah ke-3 pada rangkain c'
5. Menentukan posisi penyimpanan data pada rangkaian a, berdasarkan data pada rangkaian c'
6. Menyimpan data hasil langkah ke-5 pada rangkaian b

Berdasarkan langkah algoritma *counting sort* tersebut, maka perulangan *for* pada baris ke-9 sampai ke-10, merupakan implementasi pada langkah ke ....

```
1 void sort(char arr[])
2 {
3     int n = arr.length;
4     char output[] = new char[n];
5     int count[] = new int[256];
6
7     .....
8
9     for (int i = 0; i < n; ++i)
10        arr[i] = output[i];
11
12    .....
13 }
```

- A. 3
- B. 4
- C. 5
- D. 6

Penulisan deklarasi yang benar untuk *array* dua dimensi yang bertipe *string*, dengan nama *array* "kata" adalah ....

- A. string [ ][ ] kata;
- B. String = [ ][ ] kata;
- C. String [ ][ ] kata;
- D. string = [ ][ ] kata;

Penulisan deklarasi yang benar untuk *array linked list* dengan nama variabel topi adalah

....

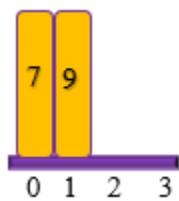
- A. List : new topi ArrayList();
- B. List topi = new ArrayList();
- C. List = new topi ArrayList();
- D. List topi: new ArrayList();

Berdasarkan tabel operasi pada *dequeue* berikut, isi kolom *Deque Contents* pada baris ke-1 adalah ....

No	Method	Return Value	Deque Contents
1	addLast(5)	-	
2	addFirst(3)	-	
3	addFirst(7)	-	
4	first( )	7	

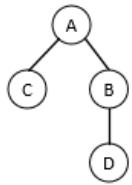
- A. 1
- B. 3
- C. 5
- D. 7

Berdasarkan ilustrasi gambar proses *queue* berikut, elemen 7 berada di indeks ke-0 *queue* dan elemen 9 berada di indeks ke-1. Jika ingin menghapus elemen awal *queue*, maka perintah ADT yang diberikan adalah ....

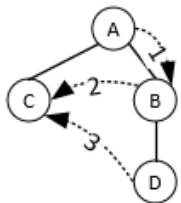


- A. enqueue()
- B. enqueue(7)
- C. dequeue()
- D. dequeue(7)

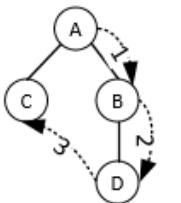
Berdasarkan *graph* berikut, jika dilakukan pencarian menggunakan algoritma *dept first search*, maka langkah penelusuran yang tepat adalah ....



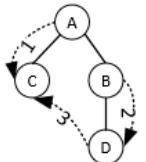
A. Pilihan A



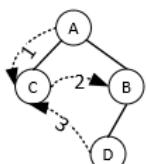
B. Pilihan B



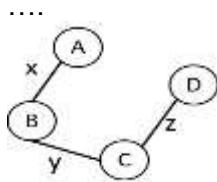
C. Pilihan C



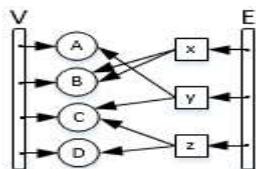
D. Pilihan D



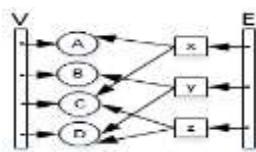
Berdasarkan graph dibawah ini representasi dalam *edge list structure* yang tepat adalah



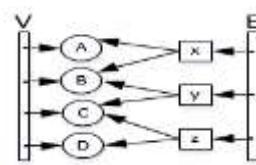
A. Pilihan A



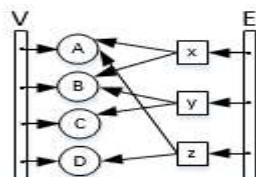
B. Pilihan B



C. Pilihan C



D. Pilihan D



*Output* dari coding berikut adalah ....

```
1 class tipe{  
2     public static void main(String[] args) {  
3         int x = 5;  
4         int y = 15;  
5         int z = x + y;  
6         System.out.println("Nilai z = " + z);  
7     }  
8 }
```

- A. Nilai z = 10
- B. Nilai z = 15
- C. Nilai z = 20
- D. Nilai z = 25

Output dari potongan pada coding DFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(5);
    g.addEdge(1, 2);
    g.addEdge(1, 4);
    g.addEdge(2, 4);
    g.addEdge(4, 1);
    g.addEdge(4, 3);
    g.addEdge(3, 3);
    g.DFS(4);
}
```

- A. 4 3 1 2
- B. 4 1 2 3
- C. 4 2 3 1
- D. 4 1 3 2

Berdasarkan ilustrasi gambar proses *dequeue* berikut, elemen 7 berada di indeks ke-0 antrian dan elemen 3 berada di indeks ke-1 antrian. Jika ingin menghapus elemen 3, maka perintah ADT yang diberikan adalah ....



- A. removeLast():
- B. removeLast()
- C. RemoveLast();
- D. RemoveLast().

Nilai antara *vertex* yang *adjacent* disebut dengan ....

- A. *path*
- B. *weight*
- C. *edge*
- D. *note*

Berdasarkan ilustrasi *stack array* berikut, jika ingin menambahkan elemen e pada indeks ke-4, maka menggunakan perintah ADT ....

b	r	e	d				
0	1	2	3	4	5	6	7

- A. Add(3, e)
- B. Add(4, e)
- C. add(3, e)
- D. add(4, e)

Berikut terdapat coding *method sort* dengan algoritma *merge sort*, pada baris ke-5 terdapat kesalahan penulisan coding yaitu pada simbol ....

```
1 void sort(int arr[], int l, int r)
2 {
3     if (l < r)
4     {
5         int m : (l+r)/2;
6
7         sort(arr, l, m);
8         sort(arr , m+1, r);
9
10        merge(arr, l, m, r);
11    }
12 }
```

- A. ;
- B. /
- C. :
- D. +

Penulisan deklarasi *coding* yang benar untuk tipe data *string* adalah ....

- A. string nama saya = "nama";
- B. string nama : "nama saya";
- C. String nama = "nama saya";
- D. String nama saya : "nama";

Ilustrasi berikut merupakan rangkaian *array* berdimensi ....

e <sub>0</sub>	e <sub>1</sub>	e <sub>2</sub>	e <sub>3</sub>	e <sub>4</sub>
----------------	----------------	----------------	----------------	----------------

- A. satu
- B. dua
- C. tiga
- D. empat

Pada kompleksitas komputasi, adanya struktur data seharusnya proses komputasi menggunakan memori sekecil mungkin, yang dikenal dengan istilah ... *complexity*.

- A. *space*
- B. *running*
- C. *time*
- D. *correctness*

Berdasarkan data berikut (2, 4, 1, 29, 47, 5, 9, 3), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan yaitu memproses data tersebut menjadi ....

- A. (1, 29, 47, 5) dan (2, 4, 9, 3)
- B. (4, 1, 29, 47) dan (2, 5, 9, 3)
- C. (2, 4, 1, 29) dan (47, 5, 9, 3)
- D. (2, 4, 1, 47) dan (29, 5, 9, 3)

Algoritma berikut yang termasuk ke dalam *comparison-based sorting* adalah ....

- A. *low-sort*
- B. *quick-sort*
- C. *counting-sort*
- D. *radix-sort*

Peserta pelatihan menulis, dapat melakukan pencarian pada halaman website pelatihan, dengan cara memasukkan kata atau kalimat pada kolom yang disediakan. Kata atau kalimat tersebut pada proses searching dikenal dengan istilah ....

- A. *edge*
- B. *path*
- C. *kata kunci*
- D. *rule*

Berdasarkan tabel proses *stack* berikut, *return value* perintah *pop()* pada baris ke-3 adalah ....

No	Method	Return Value	Stack Contents
1	push(4)	-	(4)
2	size()	<b>1</b>	(4)
3	pop()	...	()
4	push(6)	-	(6)

- A. -
- B. 1
- C. 4
- D. 6

Berdasarkan ilustrasi rangkaian *linked list* berikut, yang dimaksud dengan *node head* adalah ....



- A. LAX
- B. MSP
- C. ATL
- D. BOS

Sebuah *node* yang berada di level 0 pada sebuah *Tree* disebut dengan ....

- A. *child*
- B. *subtree*
- C. *root*
- D. *path*

Berdasarkan ekspresi aritmatikanya, *array binary tree* yang tepat dari aritmatika

$$(((2 * 6) - 4) + ((7 / 3) / 7))$$

adalah ...

- A. 

+	-	/	*	4	/	7	2	6				7	3		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
- B. 

(	2	*	6	)	-	4	+	(	7	/	3	)	/	7
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
- C. 

2	*	6	-	4	+	7	/	3	/	7				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
- D. 

+	-	/	*	7	/	7	3	4			2	6		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Seorang pegawai kecamatan sedang memasukkan nama penduduk yang berada di Kecamatan tersebut dengan menggunakan seperangkat komputer. Berdasarkan contoh kasus tersebut, data yang sedang diproses adalah ....

- A. Pegawai
- B. Kecamatan
- C. Memasukkan
- D. Nama penduduk

Berdasarkan rangkain Data  $a$  berikut (3, 1, 2, 5, 4, 3, 0, 2), jika akan dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka berdasarkan proses algoritma *counting-sort* Morin (2012), rangaian Data  $c$  adalah ....

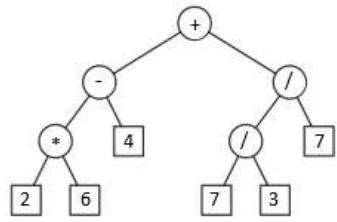
- A. (1, 1, 1, 2, 2, 2)
- B. (0, 1, 2, 3, 4, 5)
- C. (1, 1, 2, 2, 1, 1)
- D. (0, 1, 2, 2, 3, 3, 4, 5)

Berdasarkan tabel operasi *queue* berikut, isi dari *queue contents* baris ke-4 adalah ....

No	Method	Return Value	Queue Contents
1	enqueue (3)	-	
2	enqueue (9)	-	
3	first ()	3	
4	enqueue (8)	-	

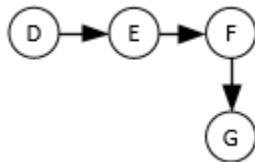
- A. 3, 9, 8
- B. 3, 8, 9
- C. 3, 8
- D. 3, 9

Berdasarkan representasiakan *binary tree* berikut, ekspresi aritmatikanya yang tepat adalah ...



- A.  $((4 / 2) + 6) - ((3 + 7) * 7)$
- B.  $((6 - 2) / 4) * ((7 * 7) - 3)$
- C.  $((2 / 4) + 6) + ((7 * 3) * 7)$
- D.  $((2 * 6) - 4) + ((7 / 3) / 7)$

Berdasarkan graph dibawah ini representasi dalam *matriks* yang tepat adalah ....



A. Pilihan A

	D	E	F	G
D	0	0	0	0
E	0	1	0	0
F	0	0	1	0
G	0	0	0	1

B. Pilihan B

	D	E	F	G
D	0	1	0	0
E	0	0	1	0
F	0	0	0	1
G	0	0	0	0

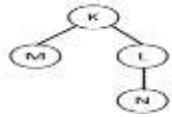
C. Pilihan C

	D	E	F	G
D	0	0	0	0
E	1	0	0	0
F	0	1	0	0
G	0	0	1	0

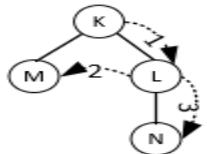
D. Pilihan D

	D	E	F	G
D	1	0	0	0
E	0	1	0	0
F	0	0	1	0
G	0	0	0	0

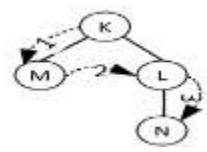
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran pada *graph* berikut adalah ....



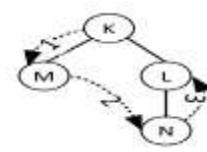
A. Pilihan A



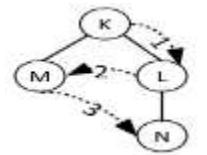
B. Pilihan B



C. Pilihan C



D. Pilihan D



Output dari potongan pada coding BFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(8);
    g.addEdge(4, 5);
    g.addEdge(4, 6);
    g.addEdge(5, 6);
    g.addEdge(6, 4);
    g.addEdge(6, 7);
    g.addEdge(7, 7);
    g.BFS(6);
}
```

- A. 6 7 5 4
- B. 6 5 4 7
- C. 6 4 5 7
- D. 6 4 7 5

Berikut merupakan langkah-langkah mengurutkan data dari terkecil ke terbesar dengan menggunakan algoritma *counting-sort*:

1. Menghitung banyaknya data yang sama pada rangkaian data awal (rangkaian a)
2. Menyimpan data hasil langkah ke-1 pada rangkain c
3. Menjumlahkan data yang terdapat pada rangkaian c
4. Menyimpan data hasil langkah ke-3 pada rangkain c'
5. Menentukan posisi penyimpanan data pada rangkaian a, berdasarkan data pada rangkaian c'
6. Menyimpan data hasil langkah ke-5 pada rangkaian b

Berdasarkan langkah-langkah diatas, maka perulangan *for* pada baris ke-9 sampai ke-10, merupakan implementasi dari langkah ke ....

```
1 void sort(char arr[])
2 {
3     int n = arr.length;
4     char output[] = new char[n];
5     int count[] = new int[256];
6
7     .....
8
9     for (int i = 1; i <= 255; ++i)
10        count[i] += count[i - 1];
11
12     .....
13 }
```

- A. 1
- B. 2
- C. 3
- D. 5

Penulisan deklarasi yang benar *array* satu dimensi yang bertipe *String*, dengan nama *array negara*, adalah ....

- A. String [ ] negara = new String[2];
- B. String [ ] = negara;
- C. string [ ] negara = new String[2];
- D. string [ ] = negara;

Penulisan deklarasi *linked list* dengan nama variabel bola yang benar adalah ....

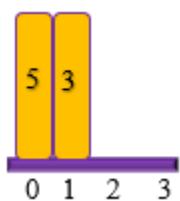
- A. LinkedList bola : new LinkedList();
- B. LinkedList bola = new LinkedList();
- C. LinkedList : new bola LinkedList();
- D. LinkedList = new bola LinkedList();

Berdasarkan tabel proses *deque* berikut, elemen yang terdapat pada kolom *Deque Contents* baris ke-4 adalah ....

No	Method	Return Value	Deque Contents
1	addFirst(6)	-	(6)
2	last( )	6	(6)
3	addFirst(8)	-	(8, 6)
4	isEmpty( )	False	( ... )
5	last( )	6	(8, 6)

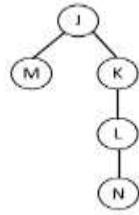
- A. (8, 6)
- B. (8)
- C. (6)
- D. ( )

Berdasarkan ilustrasi gambar proses *queue* berikut, elemen 5 berada di indeks ke-0 *queue*. Jika akan menambahkan elemen 3 berada pada akhir antrian (indek ke-1), maka perintah ADT yang diberikan adalah ....

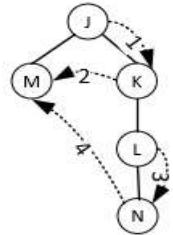


- A. enqueue(3)
- B. dequeue(1)
- C. enqueue(1)
- D. dequeue(3)

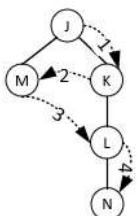
Berdasarkan *graph* berikut, jika dilakukan pencarian menggunakan algoritma *dept first search*, maka langkah penelusuran yang tepat adalah ....



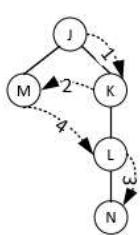
A. Pilihan A



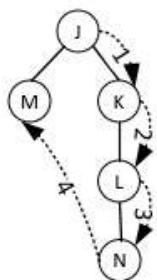
B. Pilihan B



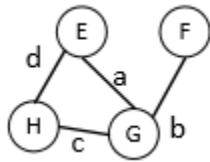
C. Pilihan C



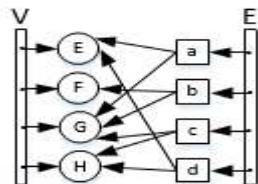
D. Pilihan D



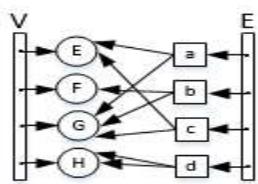
Berdasarkan graph dibawah ini representasi dalam *edge list structure* yang tepat adalah



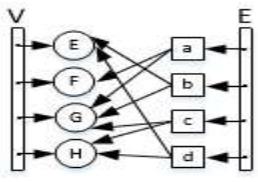
A. Pilihan A



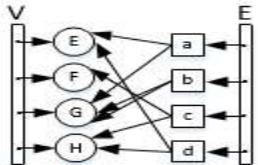
B. Pilihan B



C. Pilihan C



D. Pilihan D



*Output* dari coding berikut adalah ....

```
1 class tipe {  
2     public static void main(String[] args) {  
3         int a = 20;  
4         boolean hasil = (a = 15);  
5         System.out.println("Output coding adalah : " + hasil);  
6     }  
7 }
```

- A. Output coding adalah : 15
- B. Output coding adalah : 20
- C. Output coding adalah : true
- D. Output coding adalah : false

Output dari potongan pada coding DFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(8);
    g.addEdge(4, 5);
    g.addEdge(4, 6);
    g.addEdge(5, 6);
    g.addEdge(6, 4);
    g.addEdge(6, 7);
    g.addEdge(7, 7);
    g.DFS(6);
}
```

- A. 6 5 7 4
- B. 6 7 4 5
- C. 6 5 4 7
- D. 6 4 5 7

Berdasarkan ilustrasi gambar proses *dequeue* berikut, elemen 7 berada di indeks ke-0 antrian dan elemen 3 berada di indeks ke-1 antrian. Jika ingin menghapus elemen 7, maka perintah ADT yang diberikan adalah ....



- A. removeFirst()
- B. removeFirst(0)
- C. removeFirst(3)
- D. removeFirst(7)

Garis yang menghubungkan setiap *vertex* pada *graph* disebut dengan ....

- A. *weight*
- B. *path*
- C. *vertex*
- D. *edge*

b	r	e	d			
---	---	---	---	--	--	--

Berdasarkan ilustrasi *stack array* di atas, jika ingin menghapus elemen "r", maka perintah ADT yang tepat adalah ....

- A. remove(1)
- B. removes(1)
- C. Removes(1)
- D. move(1)

Berikut adalah coding *method sort* (baris ke 7) pada algoritma *merge sort*. Terdapat kesalahan penulisan coding yaitu pada symbol ....

```
1 void sort(int arr[], int l, int r)
2 {
3     if (l < r)
4     {
5         int m = (l+r)/2;
6
7         sort("arr, l, m);
8         sort(arr , m+1, r);
9
10        merge(arr, l, m, r);
11    }
12 }
```

- A. (
- B. ;
- C. ,
- D. "

Penulisan deklarasi coding yang benar untuk tipe data *integer* adalah ....

- A. int luas;
- B. Int Luas;
- C. int luas:
- D. Int Luas:

Isi array pada indeks ke-3 sesuai ilustrasi berikut adalah ....

Australia	Ceko	Jepang	Canada
-----------	------	--------	--------

- A. Australia
- B. Ceko
- C. Jepang
- D. Canada

Operasi struktur data diusahakan memiliki nilai *running time* sekecil mungkin yang dikenal sebagai ... *complexity*.

- A. *correctness*
- B. *running*
- C. *space*
- D. *time*

Berdasarkan data (8, 4, 78, 5, 2, 10, 6, 2), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan untuk mengurutkan data yaitu memproses data tersebut menjadi ....

- A. (8, 4, 78, 5) dan (2, 10, 6, 2)
- B. (4, 78, 5, 2) dan (8, 10, 6, 2)
- C. (78, 5, 2, 10) dan (6, 2, 8, 4)
- D. (5, 2, 10, 6) dan (8, 4, 78, 2)

Penyusunan data secara terurut dari nilai terbesar hingga terkecil, disebut dengan ....

- A. *sorting*
- B. *searching*
- C. *descending*
- D. *ascending*

Seseorang sedang melakukan pencarian nama kontak pada ponselnya, dengan memasukkan kata yaitu nama orang yang sedang dicari. Penulisan kata/nama orang tersebut pada proses *searching* dikenal dengan istilah ....

- A. *edge*
- B. *lock*
- C. *low*
- D. *key*

Berdasarkan tabel operasi *stack* berikut, isi *Stack Contents* pada baris ke-4 adalah ....

No	Method	Return Value	Stack Contents
1	push(6)	-	
2	push(7)	-	
3	size()	2	
4	pop()	7	

- A. (6)
- B. (7)
- C. (6, 7)
- D. (7, 6)

Penulisan deklarasi *linked list* dengan nama variabel bola yang benar adalah ....

- A. LinkedList bola : new LinkedList();
- B. LinkedList bola = new LinkedList();
- C. LinkedList : new bola LinkedList();
- D. LinkedList = new bola LinkedList();

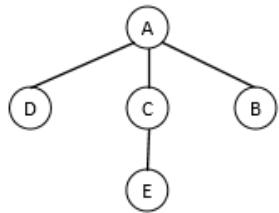
Berdasarkan data berikut (8, 4, 20, 8, 45, 1, 6, 9), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan yaitu memproses data tersebut menjadi ....

- A. (20, 8, 45, 1) dan (8, 4, 6, 9)
- B. (8, 4, 20, 8) dan (45, 1, 6, 9)
- C. (8, 4, 20, 45) dan (8, 1, 6, 9)
- D. (1, 6, 9, 8) dan (4, 20, 8, 45)

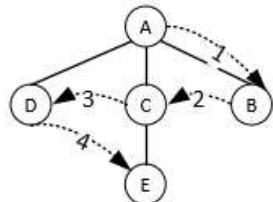
Berdasarkan rangkain Data  $a$  berikut (3, 1, 2, 5, 4, 3, 0, 2), jika akan dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka berdasarkan proses algoritma *counting-sort* Morin (2012), rangaian Data  $c$  adalah ....

- A. (1, 1, 1, 2, 2, 2)
- B. (0, 1, 2, 3, 4, 5)
- C. (1, 1, 2, 2, 1, 1)
- D. (0, 1, 2, 2, 3, 3, 4, 5)

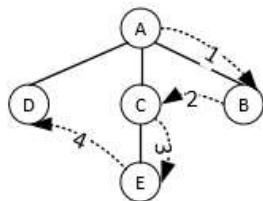
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran yang tepat adalah ....



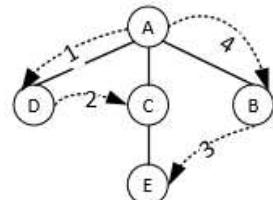
A. Pilihan A



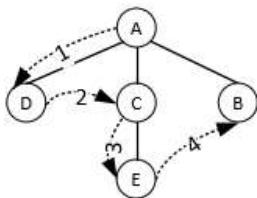
B. Pilihan B



C. Pilihan C



D. Pilihan D



Berdasarkan tabel operasi *stack* berikut, isi *Stack Contents* pada baris ke-4 adalah ....

No	Method	Return Value	Stack Contents
1	push(6)	-	
2	push(7)	-	
3	size()	2	
4	pop()	7	

- A. (6)
- B. (7)
- C. (6, 7)
- D. (7, 6)

Penulisan deklarasi *linked list* dengan nama variabel bola yang benar adalah ....

- A. LinkedList bola : new LinkedList();
- B. LinkedList bola = new LinkedList();
- C. LinkedList : new bola LinkedList();
- D. LinkedList = new bola LinkedList();

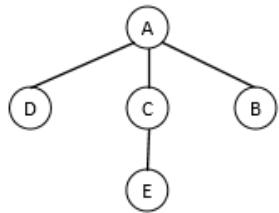
Berdasarkan data berikut (8, 4, 20, 8, 45, 1, 6, 9), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan yaitu memproses data tersebut menjadi ....

- A. (20, 8, 45, 1) dan (8, 4, 6, 9)
- B. (8, 4, 20, 8) dan (45, 1, 6, 9)
- C. (8, 4, 20, 45) dan (8, 1, 6, 9)
- D. (1, 6, 9, 8) dan (4, 20, 8, 45)

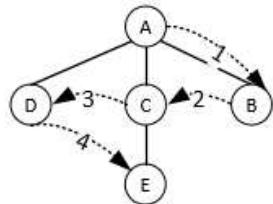
Berdasarkan rangkain Data  $a$  berikut (3, 1, 2, 5, 4, 3, 0, 2), jika akan dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka berdasarkan proses algoritma *counting-sort* Morin (2012), rangaian Data  $c$  adalah ....

- A. (1, 1, 1, 2, 2, 2)
- B. (0, 1, 2, 3, 4, 5)
- C. (1, 1, 2, 2, 1, 1)
- D. (0, 1, 2, 2, 3, 3, 4, 5)

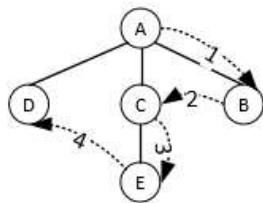
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran yang tepat adalah ....



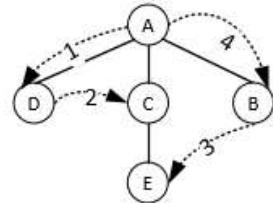
A. Pilihan A



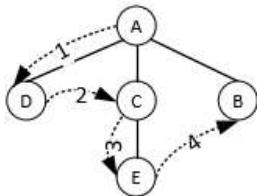
B. Pilihan B



C. Pilihan C



D. Pilihan D



Penyusunan data secara terurut dari nilai terbesar hingga terkecil, disebut dengan ....

- A. Sorting
- B. Searching
- C. Descending
- D. Ascending

Berdasarkan ilustrasi gambar proses *dequeue* berikut, jika ingin menambahkan elemen 3 pada indeks ke-2, maka perintah ADT yang diberikan adalah ....



- A. addFirst()
- B. addFirst(0)
- C. addFirst(3)
- D. addFirst(5)

Output dari potongan pada coding DFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(8);
    g.addEdge(4, 5);
    g.addEdge(4, 6);
    g.addEdge(5, 6);
    g.addEdge(6, 4);
    g.addEdge(6, 7);
    g.addEdge(7, 7);
    g.DFS(6);
}
```

- A. 6 5 7 4
- B. 6 7 4 5
- C. 6 5 4 7
- D. 6 4 5 7

Berdasarkan ilustrasi *stack array* berikut, jika ingin menambahkan elemen r pada indeks ke-2, maka menggunakan perintah ADT ....

b	r	e	d				
0	1	2	3	4	5	6	7

- A. add(2, r)
- B. add(3, r)
- C. Add(2, r)
- D. Add(3, r)

Berdasarkan tabel operasi *queue* berikut, isi dari *queue contents* baris ke-4 adalah ....

No	Method	Return Value	Queue Contents
1	enqueue (3)	-	
2	enqueue (9)	-	
3	first ()	3	
4	enqueue (8)	-	

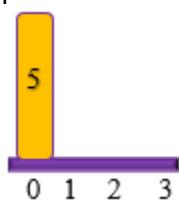
- A. 3, 9, 8
- B. 3, 8, 9
- C. 3, 8
- D. 3, 9

Berdasarkan tabel operasi pada *dequeue* berikut, isi kolom *Deque Contents* baris ke-4 adalah ....

No	Method	Return Value	Deque Contents
1	addLast(5)	-	
2	addFirst(3)	-	
3	addFirst(7)	-	
4	first( )	7	

- A. ()
- B. (5)
- C. (3, 5)
- D. (7, 3, 5)

Berdasarkan ilustrasi gambar proses *queue* berikut, jika ingin menambahkan elemen 10 pada indeks ke-1, perintah ADT yang diberikan adalah ....



- A. Enqueue(1)
- B. Enqueue(10)
- C. enqueue(1)
- D. enqueue(10)

Garis yang menghubungkan setiap *vertex* pada *graph* disebut dengan ....

- A. *Weight*
- B. *Path*
- C. *Vertex*
- D. *Edge*

Berdasarkan ekspresi aritmatikanya, *array binary tree* yang tepat dari aritmatika

$$(((2 * 6) - 4) + ((7 / 3) / 7))$$

adalah ...

- A. 

+	-	/	*	4	/	7	2	6				7	3		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
- B. 

(	2	*	6	)	-	4	+	(	7	/	3	)	/	7
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
- C. 

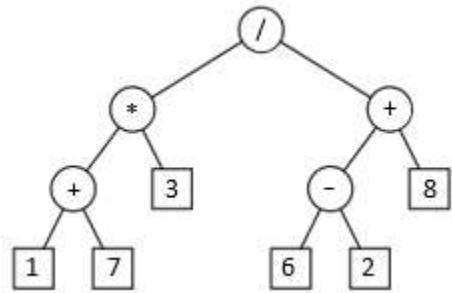
2	*	6	-	4	+	7	/	3	/	7				
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
- D. 

+	-	/	*	7	/	7	3	4			2	6		
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Sebuah *node* yang berada di level 0 pada sebuah *Tree* disebut dengan ....

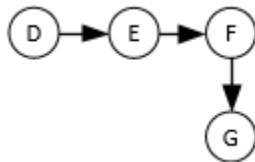
- A. *Child*
- B. *Subtree*
- C. *Root*
- D. *Path*

Berdasarkan representasiakan *binary tree* berikut, ekspresi aritmatikanya yang tepat adalah ...



- A.  $((1 + 7) * 3) / ((6 - 2) + 8)$
- B.  $((3 * 1) - 7) + ((6 / 2) / 8)$
- C.  $((7 / 1) + 3) - ((2 + 6) * 8)$
- D.  $((1 - 7) * 3) * ((2 * 6) - 8)$

Berdasarkan graph dibawah ini representasi dalam *matriks* yang tepat adalah ....



A. Pilihan A

	D	E	F	G
D	0	0	0	0
E	0	1	0	0
F	0	0	1	0
G	0	0	0	1

B. Pilihan B

	D	E	F	G
D	0	1	0	0
E	0	0	1	0
F	0	0	0	1
G	0	0	0	0

C. Pilihan C

	D	E	F	G
D	0	0	0	0
E	1	0	0	0
F	0	1	0	0
G	0	0	1	0

D. Pilihan D

	D	E	F	G
D	1	0	0	0
E	0	1	0	0
F	0	0	1	0
G	0	0	0	0

Berikut merupakan langkah-langkah mengurutkan data dari terkecil ke terbesar dengan menggunakan algoritma *counting-sort*:

1. Menghitung banyaknya data yang sama pada rangkaian data awal (rangkaian a)
2. Menyimpan data hasil langkah ke-1 pada rangkain c
3. Menjumlahkan data yang terdapat pada rangkaian c
4. Menyimpan data hasil langkah ke-3 pada rangkain c'
5. Menentukan posisi penyimpanan data pada rangkaian a, berdasarkan data pada rangkaian c'
6. Menyimpan data hasil langkah ke-5 pada rangkaian b

Berdasarkan langkah-langkah diatas, maka perulangan *for* pada baris ke-9 sampai ke-10, merupakan implementasi dari langkah ke ....

```
1 void sort(char arr[])
2 {
3     int n = arr.length;
4     char output[] = new char[n];
5     int count[] = new int[256];
6
7     .....
8
9     for (int i = 1; i <= 255; ++i)
10        count[i] += count[i - 1];
11
12     .....
13 }
```

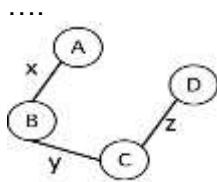
- A. 1
- B. 2
- C. 3
- D. 5

Berikut terdapat coding *method sort* dengan algoritma *merge sort*, pada baris ke-5 terdapat kesalahan penulisan coding yaitu pada simbol ....

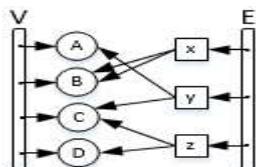
```
1 void sort(int arr[], int l, int r)
2 {
3     if (l < r)
4     {
5         int m : (l+r)/2;
6
7         sort(arr, l, m);
8         sort(arr , m+1, r);
9
10        merge(arr, l, m, r);
11    }
12 }
```

- A. ;
- B. /
- C. :
- D. +

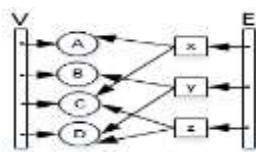
Berdasarkan graph dibawah ini representasi dalam *edge list structure* yang tepat adalah



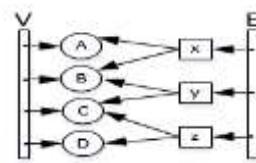
A. Pilihan A



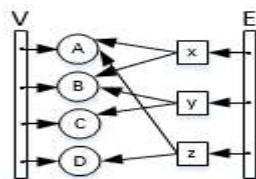
B. Pilihan B



C. Pilihan C



D. Pilihan D



Output dari potongan pada coding BFS berikut adalah ....

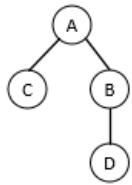
```
public static void main(String args[])
{
    Graph g = new Graph(4);
    g.addEdge(1, 0);
    g.addEdge(1, 3);
    g.addEdge(0, 3);
    g.addEdge(3, 1);
    g.addEdge(3, 2);
    g.addEdge(2, 2);
    g.BFS(3);
}
```

- A. 3 2 0 1
- B. 3 1 2 0
- C. 3 0 2 1
- D. 3 2 1 0

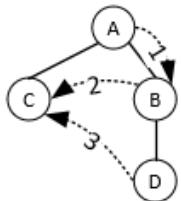
Pada proses *searching metode graph*, vertex yang dicari dapat disebut dengan ....

- A. *Lock*
- B. *Key*
- C. *Weight*
- D. *Path*

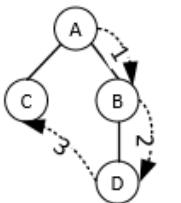
Berdasarkan *graph* berikut, jika dilakukan pencarian menggunakan algoritma *dept first search*, maka langkah penelusuran yang tepat adalah ....



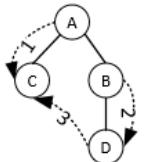
A. Pilihan A



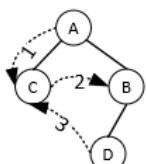
B. Pilihan B



C. Pilihan C



D. Pilihan D



*Output* dari coding berikut adalah ....

```
1 class tipe{  
2     public static void main(String[] args) {  
3         int x = 5;  
4         int y = 15;  
5         int z = x + y;  
6         System.out.println("Nilai z = " + z);  
7     }  
8 }
```

- A. Nilai z = 10
- B. Nilai z = 15
- C. Nilai z = 20
- D. Nilai z = 25

Penulisan deklarasi *array* satu dimensi yang bertipe *integer*, dengan nama *array nomor*, adalah ....

- A. Int [ ] nomor = new int[2];
- B. Int [ ] = nomor;
- C. int [ ] nomor = new int[2];
- D. int [ ] = nomor;

Berdasarkan ilustrasi rangkaian *linked list* berikut, yang dimaksud dengan *node tail* adalah ....



- A. LAX
- B. MSP
- C. ATL
- D. BOS

Ilustrasi berikut merupakan rangkaian *array* berdimensi ....

e <sub>0</sub>	e <sub>1</sub>	e <sub>2</sub>	e <sub>3</sub>	e <sub>4</sub>
----------------	----------------	----------------	----------------	----------------

- A. Satu
- B. Dua
- C. Tiga
- D. Empat

Seorang pegawai kecamatan sedang memasukkan nama penduduk yang berada di Kecamatan tersebut dengan menggunakan seperangkat komputer. Berdasarkan contoh kasus tersebut, data yang sedang diproses adalah ....

- A. Pegawai
- B. Kecamatan
- C. Memasukkan
- D. Nama penduduk

Penulisan deklarasi nilai bertipe data *null* yang benar adalah ....

- A. Integer nilai = null
- B. Int nilai : null;
- C. string huruf : null
- D. String nilai = null;

Pada kompleksitas komputasi, adanya struktur data seharusnya proses komputasi menggunakan memori sekecil mungkin, yang dikenal dengan istilah ... *complexity*.

- A. *Space*
- B. *Running*
- C. *Time*
- D. *Correctness*

Berdasarkan tabel operasi *stack* berikut, isi *stack contents* pada baris ke-4 adalah ....

No	Method	Return Value	Stack Contents
1	push (7)	-	
2	push (9)	-	
3	top ()	9	
4	push (4)	-	

- A. (4)
- B. (7, 9)
- C. (7, 9, 4)
- D. (4, 7, 9)

Penulisan deklarasi *linked list* dengan nama variabel kotak yang benar adalah ....

- A. LinkedList kotak : new LinkedList();
- B. LinkedList : new kotak LinkedList();
- C. LinkedList kotak = new LinkedList();
- D. LinkedList = new kotak LinkedList();

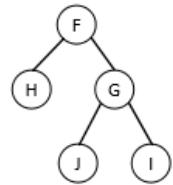
Berdasarkan data berikut (2, 4, 1, 29, 47, 5, 9, 3), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan yaitu memproses data tersebut menjadi ....

- A. (1, 29, 47, 5) dan (2, 4, 9, 3)
- B. (4, 1, 29, 47) dan (2, 5, 9, 3)
- C. (2, 4, 1, 29) dan (47, 5, 9, 3)
- D. (2, 4, 1, 47) dan (29, 5, 9, 3)

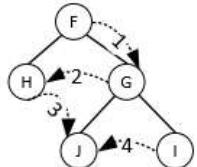
Berdasarkan rangkain Data  $a$  berikut (2, 4, 3, 2, 5, 1, 4, 0), jika akan dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka berdasarkan proses algoritma *counting-sort* Morin (2012), rangaian Data  $c$  adalah ....

- A. (1, 1, 2, 1, 2, 1)
- B. (0, 1, 2, 2, 3, 4, 4, 5)
- C. (0, 1, 2, 3, 4, 5)
- D. (2, 2, 2, 1, 1, 1)

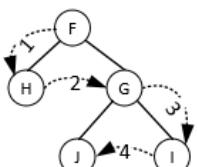
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran yang tepat adalah ...



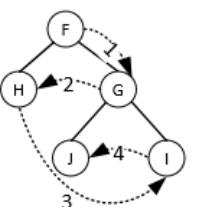
A. Pilihan A



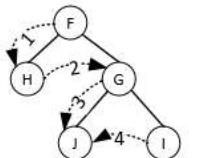
B. Pilihan B



C. Pilihan C



D. Pilihan D



Penyusunan data secara terurut dari nilai terkecil hingga terbesar, disebut dengan ....

- A. Ascending
- B. Descending
- C. Searching
- D. Sorting

Berdasarkan ilustrasi gambar proses *dequeue* berikut, jika ingin menambahkan elemen 12 pada indeks ke-1, maka perintah ADT yang diberikan adalah ....



- A. addFirst()
- B. addFirst(3)
- C. addFirst(5)
- D. addFirst(12)

Output dari potongan pada coding DFS berikut adalah ....

```
public static void main(String args[])
{
    Graph g = new Graph(5);
    g.addEdge(1, 2);
    g.addEdge(1, 4);
    g.addEdge(2, 4);
    g.addEdge(4, 1);
    g.addEdge(4, 3);
    g.addEdge(3, 3);
    g.DFS(4);
}
```

- A. 4 3 1 2
- B. 4 1 2 3
- C. 4 2 3 1
- D. 4 1 3 2

Berdasarkan ilustrasi *stack array* berikut, jika ingin menambahkan elemen e pada indeks ke-4, maka menggunakan perintah ADT ....

b	r	e	d				
0	1	2	3	4	5	6	7

- A. Add(3, e)
- B. Add(4, e)
- C. add(3, e)
- D. add(4, e)

Berdasarkan tabel operasi *queue* berikut, isi dari *queue contents* baris ke-4 adalah ....

No	Method	Return Value	Queue Contents
1	enqueue(10)	-	
2	enqueue(8)	-	
3	size()	2	
4	dequeue()	10	

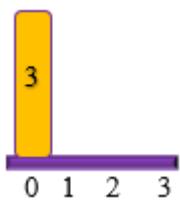
- A. -
- B. 2
- C. 8
- D. 10

Berdasarkan tabel operasi pada *dequeue* berikut, isi kolom *Deque Contents* pada baris ke-1 adalah ....

No	Method	Return Value	Deque Contents
1	addLast(5)	-	
2	addFirst(3)	-	
3	addFirst(7)	-	
4	first( )	7	

- A. 1
- B. 3
- C. 5
- D. 7

Berdasarkan ilustrasi gambar proses *queue* berikut, jika ingin menambahkan elemen 6 pada indeks ke-1, perintah ADT yang diberikan adalah ....



- A. enqueue(6)
- B. enqueue(1);
- C. Enqueue(6);
- D. Enqueue(1)

Sebuah titik yang terdapat pada *graph* disebut dengan ....

- A. *Vertex*
- B. *Edge*
- C. *Path*
- D. *Cycle*

Berdasarkan ekspresi aritmatikanya, *array binary tree* yang tepat dari aritmatika

$$(((3 / 5) + 5) - ((8 + 4) * 6))$$

adalah ....

- A. 

3	/	5	+	5	-			8	+	4	*	6			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
- B. 

/	+	-	+	*	3	5	5	8	4	6					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
- C. 

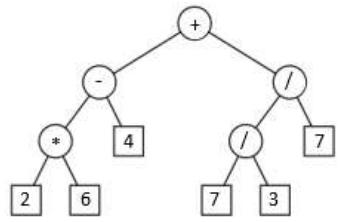
-	+	*	/	5	+	6	3	5			8	4			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
- D. 

3	/	5	+	5	-	8	+	4	*	6					
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	

Setiap *node* pada *tree* yang dapat memiliki lebih dari 1 (satu) *edge* yang menghubungkan dengan *node* dibawahnya. *Node* dibawahnya disebut dengan ....

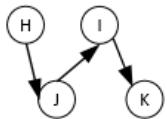
- A. *Path*
- B. *Parent*
- C. *Leaf*
- D. *Child*

Berdasarkan representasiakan *binary tree* berikut, ekspresi aritmatikanya yang tepat adalah ...



- A.  $((4 / 2) + 6) - ((3 + 7) * 7)$
- B.  $((6 - 2) / 4) * ((7 * 7) - 3)$
- C.  $((2 / 4) + 6) + ((7 * 3) * 7)$
- D.  $((2 * 6) - 4) + ((7 / 3) / 7)$

Berdasarkan graph dibawah ini representasi dalam *matriks* yang tepat adalah ....



A. Pilihan A

	H	I	J	K
H	0	1	0	0
I	0	0	0	1
J	0	1	0	0
K	0	0	0	0

B. Pilihan B

	H	I	J	K
H	0	0	1	0
I	0	0	0	1
J	1	0	0	0
K	0	0	0	0

C. Pilihan C

	H	I	J	K
H	0	0	1	0
I	0	0	0	1
J	0	1	0	0
K	0	0	0	0

D. Pilihan D

	H	I	J	K
H	0	0	1	0
I	1	0	0	0
J	1	0	0	0
K	0	0	0	0

Berikut merupakan langkah-langkah mengurutkan data dari terkecil ke terbesar dengan menggunakan algoritma *counting-sort*:

1. Menghitung banyaknya data yang sama pada rangkaian data awal (rangkaian a)
2. Menyimpan data hasil langkah ke-1 pada rangkain c
3. Menjumlahkan data yang terdapat pada rangkaian c
4. Menyimpan data hasil langkah ke-3 pada rangkain c'
5. Menentukan posisi penyimpanan data pada rangkaian a, berdasarkan data pada rangkaian c'
6. Menyimpan data hasil langkah ke-5 pada rangkaian b

Berdasarkan langkah-langkah diatas, maka perulangan *for* pada baris ke-9 sampai ke-10, merupakan implementasi dari langkah ke ....

```
1 void sort(char arr[])
2 {
3     int n = arr.length;
4     char output[] = new char[n];
5     int count[] = new int[256];
6
7     .....
8
9     for (int i = 0; i < n; ++i)
10        ++count[arr[i]];
11
12     .....
13 }
```

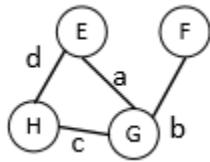
- A. 1
- B. 3
- C. 5
- D. 6

Berikut terdapat coding *method sort* dengan algoritma *merge-sort*, pada baris ke-3 terdapat kesalahan penulisan coding yaitu pada simbol ....

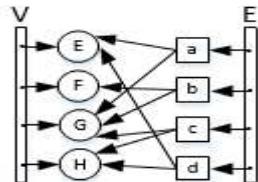
```
1 void sort(int arr[], int l, int r)
2 {
3     if (l < r);
4     {
5         int m = (l+r)/2;
6
7         sort(arr, l, m);
8         sort(arr , m+1, r);
9
10        merge(arr, l, m, r);
11    }
12 }
```

- A. (
- B. <
- C. )
- D. ;

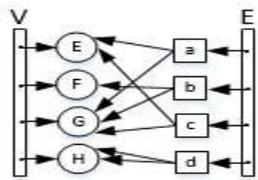
Berdasarkan graph dibawah ini representasi dalam *edge list structure* yang tepat adalah



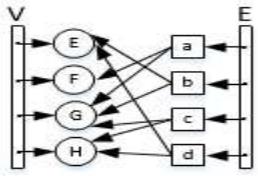
A. Pilihan A



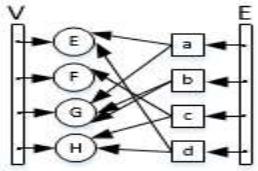
B. Pilihan B



C. Pilihan C



D. Pilihan D



Output dari potongan pada coding BFS berikut adalah ....

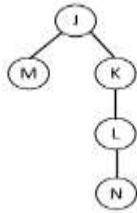
```
public static void main(String args[])
{
    Graph g = new Graph(8);
    g.addEdge(4, 5);
    g.addEdge(4, 6);
    g.addEdge(5, 6);
    g.addEdge(6, 4);
    g.addEdge(6, 7);
    g.addEdge(7, 7);
    g.BFS(6);
}
```

- A. 6 7 5 4
- B. 6 5 4 7
- C. 6 4 5 7
- D. 6 4 7 5

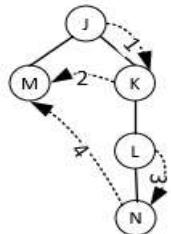
Peserta pelatihan menulis, dapat melakukan pencarian pada halaman website pelatihan, dengan cara memasukkan kata atau kalimat pada kolom yang disediakan. Kata atau kalimat tersebut pada proses searching dikenal dengan istilah ....

- A. *Edge*
- B. *Path*
- C. *Kata kunci*
- D. *Rule*

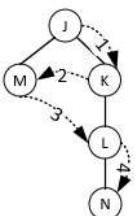
Berdasarkan *graph* berikut, jika dilakukan pencarian menggunakan algoritma *dept first search*, maka langkah penelusuran yang tepat adalah ....



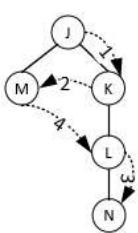
A. Pilihan A



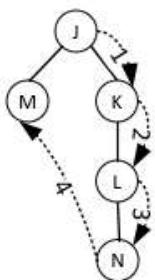
B. Pilihan B



C. Pilihan C



D. Pilihan D



*Output* dari coding berikut adalah ....

```
1 class tipe {  
2     public static void main(String[] args) {  
3         int a = 20;  
4         boolean hasil = (a = 15);  
5         System.out.println("Output coding adalah : " + hasil);  
6     }  
7 }
```

- A. Output coding adalah : 15
- B. Output coding adalah : 20
- C. Output coding adalah : true
- D. Output coding adalah : false

Penulisan deklarasi yang benar *array* satu dimensi yang bertipe *String*, dengan nama *array negara*, adalah ....

- A. String [ ] negara = new String[2];
- B. String [ ] = negara;
- C. string [ ] negara = new String[2];
- D. string [ ] = negara;

Berdasarkan ilustrasi rangkaian *linked list* berikut, yang dimaksud dengan *node head* adalah ....



- A. LAX
- B. MSP
- C. ATL
- D. BOS

Ilustrasi berikut merupakan rangkaian *array* berdimensi ....

e <sub>00</sub>	e <sub>01</sub>	e <sub>02</sub>	e <sub>03</sub>	e <sub>04</sub>
e <sub>10</sub>	e <sub>11</sub>	e <sub>12</sub>	e <sub>13</sub>	e <sub>14</sub>
e <sub>20</sub>	e <sub>21</sub>	e <sub>22</sub>	e <sub>23</sub>	e <sub>24</sub>
e <sub>30</sub>	e <sub>31</sub>	e <sub>32</sub>	e <sub>33</sub>	e <sub>34</sub>

- A. Satu
- B. Dua
- C. Tiga
- D. Empat

Seorang pengemudi taksi *online* sedang melihat arah jalan untuk menuju pemesan *online* dengan menggunakan handphone. Berdasarkan contoh kasus tersebut, data yang sedang diproses adalah ....

- A. Pengemudi
- B. Pemesan
- C. Arah jalan
- D. Handphone

Penulisan deklarasi tipe data *Boolean* yang benar adalah?

- A. boolean true : input;
- B. boolean input = false;
- C. boolean input : true;
- D. boolean false = input;

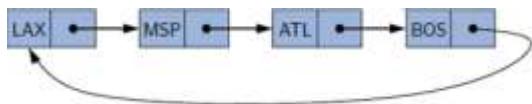
Pada kompleksitas komputasi, *Interface* struktur data harus dipastikan diimplementasikan sesuai dengan fungsinya, yang dikenal dengan istilah ....

- A. *Complexity*
- B. *Correctness*
- C. *Space*
- D. *Time*

Berdasarkan tabel operasi *stack* berikut, isi *Stack Contents* pada baris ke-4 adalah ....

No	Method	Return Value	Stack Contents
1	push(6)	-	
2	push(7)	-	
3	size()	2	
4	pop()	7	

- A. (6)
- B. (7)
- C. (6, 7)
- D. (7, 6)



Berdasarkan ilustrasi rangkaian *linked list* di atas, maka rangkaian tersebut merupakan *linked list* ....

- A. *singly*
- B. *doubly*
- C. *circularly*
- D. *manually*

Setiap *node* pada *tree* yang dapat memiliki lebih dari 1 (satu) *edge* yang menghubungkan dengan *node* dibawahnya. *Node* dibawahnya disebut dengan ....

- A. *path*
- B. *parent*
- C. *leaf*
- D. *child*

Representasi dari ekspresi aritmatika  $((4 - 4) / 6) * ((9 * 5) - 5)$  dalam *array binary tree* yang benar adalah ....

A. 

-	/	*	*	-	4	4	6	9	5	5					
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

B. 

*	/	-	-	6	*	5	4	4			9	5			
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

C. 

4	-	4	/	6	*	9	*	5			-	5			
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

D. 

4	-	4	/	6	*	9	*	5	-	5					
0	1	2	3	4	5	6	7	8	9	1	1	1	1	1	1
										0	1	2	3	4	

Penjual pada sebuah toko buku sedang menulis nama buku yang banyak dicari pembeli melalui laptop. Berdasarkan contoh kasus tersebut, yang dimaksud dengan data yang sedang diproses adalah ....

- A. penjual
- B. nama buku
- C. pembeli
- D. laptop

Berdasarkan data berikut (5, 3, 2, 4, 1, 3, 0, 4) dengan rentang data yaitu 0 sampai 5. Jika dilakukan *sorting* dengan menggunakan algoritma *counting-sort*, maka rangkaian data c yang dihasilkan ....

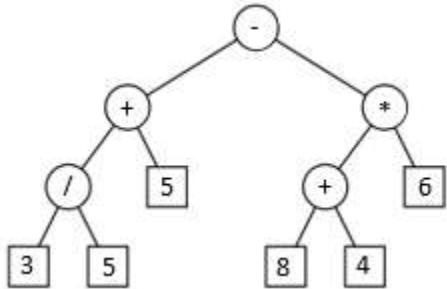
- A. (0, 1, 2, 3, 4, 5)
- B. (1, 1, 1, 2, 2, 1)
- C. (0, 1, 2, 3, 3, 4, 4, 5)
- D. (1, 1, 1, 2, 2, 2)

Berdasarkan tabel proses *queue* berikut, *return value* perintah `size()` pada baris ke-3 adalah ....

No	Method	Return Value	Queue Contents
1	<code>enqueue(1)</code>	-	(1)
2	<code>enqueue(3)</code>	-	(1, 3)
3	<code>size()</code>	...	(1, 3)
4	<code>dequeue()</code>	1	(3)

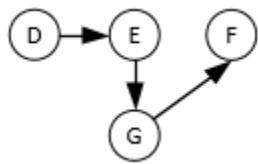
- A. -
- B. 1
- C. 2
- D. 3

Berdasarkan representasi *binary tree* berikut, ekspresi aritmatika yang tepat adalah ...



- A.  $((5 - 3) / 5) * ((6 * 4) - 8)$
- B.  $((3 / 5) + 5) - ((8 + 4) * 6)$
- C.  $((5 / 5) + 3) + ((4 * 8) * 6)$
- D.  $((3 * 5) / 5) * ((4 / 6) / 8)$

Representasi *directed graph* berikut dalam tabel yang tepat adalah ....



A. Pilihan A

D	E	F	G
E	G		F

B. Pilihan B

D	E	F	G
E	G	F	

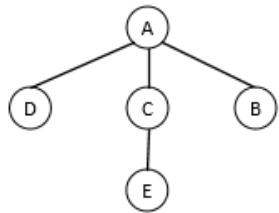
C. Pilihan C

D	E	F	G
	G	E	F

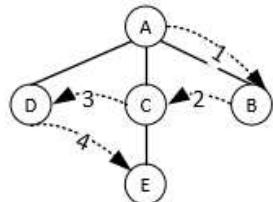
D. Pilihan D

D	E	F	G
F	G	E	

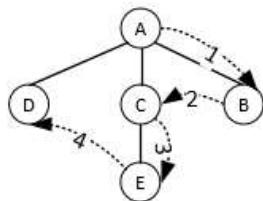
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan algoritma *breath first search*, maka langkah penelusuran yang tepat adalah ....



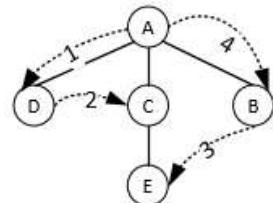
A. Pilihan A



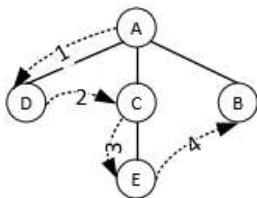
B. Pilihan B



C. Pilihan C



D. Pilihan D



Apakah *output* dari potongan *main* program pada coding BFS berikut?

```
public static void main(String args[])
{
    Graph g = new Graph(6);
    g.addEdge(2, 3);
    g.addEdge(2, 4);
    g.addEdge(3, 4);
    g.addEdge(4, 2);
    g.addEdge(4, 5);
    g.addEdge(5, 5);
    System.out.println("BFS dengan vertex awal 4");
    g.BFS(4);
}
```

- A. 4 3 2 5
- B. 4 5 3 2
- C. 4 2 5 3
- D. 4 3 5 2

Berikut terdapat langkah mengurutkan data dari terkecil ke terbesar dengan menggunakan algoritma *counting sort*.

1. Menghitung banyaknya data yang sama pada rangkaian data awal (rangkaian a)
2. Menyimpan data hasil langkah ke-1 pada rangkain c
3. Menjumlahkan data yang terdapat pada rangkaian c
4. Menyimpan data hasil langkah ke-3 pada rangkain c'
5. Menentukan posisi penyimpanan data pada rangkaian a, berdasarkan data pada rangkaian c'
6. Menyimpan data hasil langkah ke-5 pada rangkaian b

Berdasarkan langkah algoritma *counting sort* tersebut, maka perulangan *for* pada baris ke-9 sampai ke-13, merupakan implementasi pada langkah ke ....

```
1 void sort(char arr[])
2 {
3     int n = arr.length;
4     char output[] = new char[n];
5     int count[] = new int[256];
6
7     .....
8
9     for (int i = 0; i < n; ++i)
10    {
11        output[count[arr[i]] - 1] = arr[i];
12        --count[arr[i]];
13    }
14
15    .....
16 }
```

- A. 2
- B. 4
- C. 5
- D. 6

Penulisan deklarasi *array* satu dimensi yang bertipe *integer*, dengan nama *array nomor*, adalah ....

- A. Int [ ] nomor = new int[2];
- B. Int [ ] = nomor;
- C. int [ ] nomor = new int[2];
- D. int [ ] = nomor;

Penulisan deklarasi *linked list* dengan nama variabel kotak yang benar adalah ....

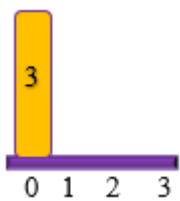
- A. LinkedList kotak : new LinkedList();
- B. LinkedList : new kotak LinkedList();
- C. LinkedList kotak = new LinkedList();
- D. LinkedList = new kotak LinkedList();

Berdasarkan tabel proses *deque* berikut, nilai pada kolom *Return Value* baris ke-2 adalah ....

No	Method	Return Value	Deque Contents
1	addFirst(8)	-	(8, 6)
2	isEmpty( )	...	(8, 6)
3	last( )	6	(8, 6)

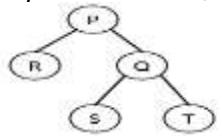
- A. True
- B. False
- C. Null
- D. Empty

Berdasarkan ilustrasi gambar proses *queue* berikut, jika ingin menambahkan elemen 6 pada indeks ke-1, perintah ADT yang diberikan adalah ....

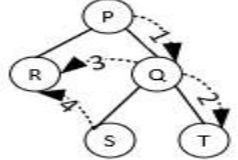


- A. enqueue(6)
- B. enqueue(1);
- C. Enqueue(6);
- D. Enqueue(1)

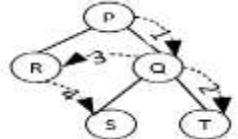
Berdasarkan *graph* berikut, jika dilakukan pencarian dengan menggunakan algoritma *dept first search*, maka tahapan menelusuri *node* pada *graph* yang tepat adalah ....



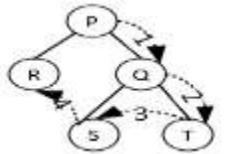
A. Pilihan A



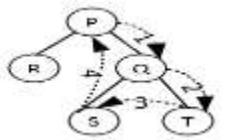
B. Pilihan B



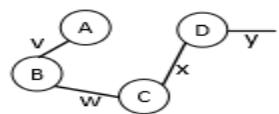
C. Pilihan C



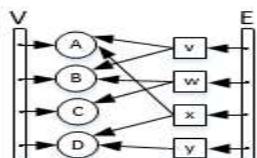
D. Pilihan D



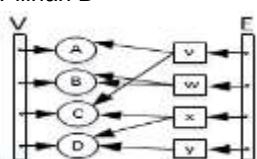
Silahkan anda representasikan *graph* berikut ke dalam *edge list structure*?



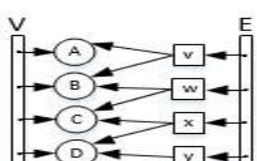
A. Pilihan A



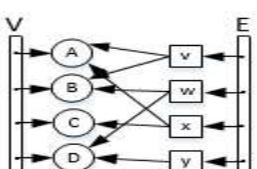
B. Pilihan B



C. Pilihan C



D. Pilihan D



Silahkan anda perhatikan coding berikut. *Output* dari coding tersebut adalah ....

```
1 class tipe{  
2     public static void main(String[] args){  
3         double x = 5.0d;  
4         double y = 2.0d;  
5         System.out.println(x + " * " + y + " = " + (x*y));  
6     }  
7 }
```

- A.  $2.0 * 5.0 = 10$
- B.  $2.0 * 5.0 = 10.0$
- C.  $5.0 * 2.0 = 10$
- D.  $5.0 * 2.0 = 10.0$

Apakah *output* dari potongan *main* program pada coding DFS berikut?

```
public static void main(String args[])
{
    Graph g = new Graph(6);
    g.addEdge(2, 3);
    g.addEdge(2, 4);
    g.addEdge(3, 4);
    g.addEdge(4, 2);
    g.addEdge(4, 5);
    g.addEdge(5, 5);
    System.out.println("DFS dengan vertex awal 4");
    g.DFS(4);
}
```

- A. 4 2 3 5
- B. 4 3 2 5
- C. 4 5 3 2
- D. 4 2 5 3

Berdasarkan ilustrasi gambar proses *dequeue* berikut, jika ingin menambahkan elemen 3 pada indeks ke-2, maka perintah ADT yang diberikan adalah ....



- A. addFirst()
- B. addFirst(0)
- C. addFirst(3)
- D. addFirst(5)

Dua buah *vertex* yang berdekatan dan memiliki *edge* antar keduanya disebut dengan ....

- A. *edge*
- B. *vertex*
- C. *adjacent*
- D. *path*

b	r	e	d			
---	---	---	---	--	--	--

Berdasarkan ilustrasi *stack array* di atas, jika ingin ditambahkan elemen “b” setelah eleman “e”, maka perintah ADT yang tepat adalah ....

- A. add(2, b)
- B. add(3, b)
- C. Add(2, b)
- D. Add(3, b)

Berikut terdapat coding *method sort* dengan algoritma *merge-sort*, pada baris ke-3 terdapat kesalahan penulisan coding yaitu pada simbol ....

```
1 void sort(int arr[], int l, int r)
2 {
3     if (l < r);
4     {
5         int m = (l+r)/2;
6
7         sort(arr, l, m);
8         sort(arr , m+1, r);
9
10        merge(arr, l, m, r);
11    }
12 }
```

- A. (
- B. <
- C. )
- D. ;

Penulisan deklarasi tipe data *Boolean* yang benar adalah ....

- A. boolean true : input;
- B. boolean input = false;
- C. boolean input : true;
- D. boolean false = input;

Ilustrasi berikut merupakan rangkaian *array* berdimensi ....

e <sub>00</sub>	e <sub>01</sub>	e <sub>02</sub>	e <sub>03</sub>	e <sub>04</sub>
e <sub>10</sub>	e <sub>11</sub>	e <sub>12</sub>	e <sub>13</sub>	e <sub>14</sub>
e <sub>20</sub>	e <sub>21</sub>	e <sub>22</sub>	e <sub>23</sub>	e <sub>24</sub>
e <sub>30</sub>	e <sub>31</sub>	e <sub>32</sub>	e <sub>33</sub>	e <sub>34</sub>

- A. satu
- B. dua
- C. tiga
- D. empat

Jenis *running times* untuk membuktikan bahwa ada operasi struktur data yang tidak sesuai target adalah ....

- A. *correctness*
- B. *amortized*
- C. *worst-case*
- D. *expected*

Berdasarkan data (4, 92, 5, 90, 2, 17, 8, 7), jika akan dilakukan *sorting* dengan menggunakan algoritma *merge-sort*, maka tahap awal yang akan dilakukan untuk mengurutkan data yaitu memproses data tersebut menjadi ....

- A. (92, 5, 90, 2) dan (4, 17, 8, 7)
- B. (4, 92, 5, 2) dan (90, 17, 8, 7)
- C. (2, 17, 8, 90) dan (4, 92, 5, 7)
- D. (4, 92, 5, 90) dan (2, 17, 8, 7)

Algoritma berikut yang termasuk ke dalam non-*comparison-based sorting* adalah ....

- A. *low-sort*
- B. *quick-sort*
- C. *counting-sort*
- D. *heap-sort*

Pada proses *searching metode graph*, vertex yang dicari dapat disebut dengan ....

- A. *lock*
- B. *key*
- C. *weight*
- D. *path*