UTM
UNIVERSITI TEKNOLOGI MALAYSIA

Sekolah Pendidikan Profesional dan Pendidikan Berterusan (SPACE)

FINAL EXAMINATION / *PEPERIKSAAN AKHIR*
SEMESTER II – SESSION 2022 / 2023
PROGRAM KERJASAMA

COURSE CODE *KOD KURSUS*	:	DDWD 2733 / DDWC 2733
COURSE NAME *NAMA KURSUS*	:	DATA STRUCTURE AND ALGORITHMS / DATA STRUCTURE *STRUKTUR DATA DAN ALGORITMA / STRUKTUR DATA*
YEAR / PROGRAMME *TAHUN / PROGRAM*	:	2 DDWD / DDWC
DURATION *TEMPOH*	:	2 HOURS 30 MINUTES *2 JAM 30 MINIT*
DATE *TARIKH*	:	JUNE / JULY 2023 *JUN / JULAI 2023*

INSTRUCTION :
ARAHAN

ANSWER **ALL QUESTIONS** IN QUESTION BOOKLET.

*JAWAB **SEMUA SOALAN DI** DALAM BUKU SOALAN.*

(You are required to write your name and your lecturer's name on your answer script)
(Pelajar dikehendaki tuliskan nama dan nama pensyarah pada skrip jawapan)

NAME / *NAMA PELAJAR*	:	.....
I.C NO. / *NO. K/PENGENALAN*	:	.....
YEAR / PROGRAMME *TAHUN / PROGRAM*	:	.....
COLLEGE NAME *NAMA KOLEJ*	:	.....
LECTURER'S NAME *NAMA PENSYARAH*	:	.....

This examination paper consists of ...17.... pages including the cover
Kertas soalan ini mengandungi ...17..... muka surat termasuk kulit hadapan

UTM UNIVERSITI TEKNOLOGI MALAYSIA | School of Professional and Continuing Education (SPACE)

PUSAT PRGORAM KERJASAMA

PETIKAN DARIPADA PERATURAN AKADEMIK ARAHAN AM – PENYELEWENGAN AKADEMIK

1. SALAH LAKU SEMASA PEPERIKSAAN

1.1. Pelajar tidak boleh melakukan mana-mana salah laku peperiksaan seperti berikut :-

1.1.1. memberi dan/atau menerima dan/atau memiliki sebarang maklumat dalam bentuk elektronik, bercetak atau apa jua bentuk lain yang tidak dibenarkan semasa berlangsungnya peperiksaan sama ada di dalam atau di luar Dewan/Bilik Peperiksaan melainkan dengan kebenaran Ketua Pengawas; atau

1.1.2. menggunakan maklumat yang diperoleh seperti di atas bagi tujuan menjawab soalan peperiksaan; atau

1.1.3. menipu atau cuba untuk menipu atau berkelakuan mengikut cara yang boleh ditafsirkan sebagai menipu semasa berlangsungnya peperiksaan; atau

1.1.4. lain-lain salah laku yang ditetapkan oleh Universiti (seperti membuat bising, mengganggu pelajar lain, mengganggu Pengawas menjalankan tugasnya).

2. HUKUMAN SALAH LAKU PEPERIKSAAN

2.1. Sekiranya pelajar didapati telah melakukan pelanggaran mana-mana peraturan peperiksaan ini, setelah diperakukan oleh Jawatankuasa Peperiksaan Fakulti dan disabitkan kesalahannya, Senat boleh mengambil tindakan dari mana-mana satu yang berikut :-

2.1.1. memberi markah SIFAR (0) bagi keseluruhan keputusan peperiksaan kursus yang berkenaan (termasuk kerja kursus); atau

2.1.2. memberi markah SIFAR (0) bagi semua kursus yang didaftarkan pada semester tersebut.

2.2. Jawatankuasa Akademik Fakulti boleh mencadangkan untuk diambil tindakan tatatertib mengikut peruntukan Akta Universiti dan Kolej Universiti, 1971, Kaedah-kaedah Universiti Teknologi Malaysia (Tatatertib Pelajar-pelajar), 1999 bergantung kepada tahap kesalahan yang dilakukan oleh pelajar.

2.3. Pelajar yang didapati melakukan kesalahan kali kedua akan diambil tindakan seperti di perkara dan dicadang untuk diambil tindakan tatatertib mengikut peruntukan Akta Universiti dan Kolej Universiti, 1971, Kaedah-kaedah Universiti Teknologi Malaysia (Tatatertib Pelajar-pelajar), 1999.

SECTION A: TRUE OR FALSE [10 MARKS]

BAHAGIAN A: BENAR ATAU SALAH [10 MARKAH]

1. Random access is not allowed in a typical implementation of Linked Lists.
Capaian rawak tidak dibenarkan dalam pelaksanaan biasa Senarai Berpaut.

2. Inserting a last **node** in the linked list share the same logic as inserting a **node** at the beginning of linked list.
*Kemasukan satu **nod** di akhir senarai berpaut berkongsi logik yang sama seperti memasukkan satu **nod** di permulaan senarai berpaut.*

3. By using an Abstract Data Type **(ADT)**, programmers can hide implementation details of program codes from the user.
*Dengan menggunakan Jenis Data Abstrak **(ADT)**, pengaturcara boleh menyembunyikan implementasi kod-kod aturcara daripada pengguna.*

4. Four basic queue operations are **Enqueue**, **Outqueue**, **Queue Front**, **Queue Rear**.
*Empat operasi asas giliran ialah **Enqueue**, **Outqueue**, **Queue Front**, **Queue Rear**.*

5. In the double linked list, the last node's link points to the first node of the list.
Dalam senarai berpaut berganda, bahagian berpaut nod yang terakhir menunjuk kepada nod yang pertama dalam senarai.

6. In a general list, data can be inserted and deleted anywhere and there are no restrictions on the operations that can be used to process the list.
Dalam senarai berjenis umum, data boleh dimasukkan dan dibuang di mana-mana dan tidak ada sekatan terhadap operasi-operasi yang boleh digunakan untuk memproses senarai.

7. The **outdegree** of a vertex in a **digraph** is the number of arcs entering the vertex.
***Luardarjah** bagi satu nod di dalam satu graf berarah adalah bilangan arka yang memasuki nod.*

8. Data cannot be sorted in descending sequence.
Data tidak boleh disusun di dalam jujukan menurun.

9. If the data in an array is sorted, binary search algorithm is more efficient than sequential search algorithm.
Jika data di dalam tatasusunan telah tersusun, algoritma carian binari lebih effektif berbanding dengan algoritma carian jujukan.

10. Traversal in a tree is a repetitive process where that algorithm calls itself.
Penjelajahan dalam pepohon adalah proses yang berulang di mana algoritma tersebut memanggil dirinya sendiri.

SECTION B : MULTIPLE CHOICES [30 MARKS]

BAHAGIAN B : ANEKA PILIHAN [30 MARKAH]

1. Which of the following is **NOT** a step in adding a **node** to a linked list?
*Manakah di antara berikut **BUKAN** satu langkah di dalam memasukkan satu **nod** ke satu senarai berpaut?*
   A. Determine the insertion point / *Memastikan titik masukkan*
   B. Point the new node to the successor */ Nod baru menunjukkan kepada nod selepas*
   C. Point the predecessor to the new node / *Nod sebelum menunjukkan kepada nod baru*
   D. Assign the head to NULL / *Tetapkan pengepala kepada NULL*

2. Which of the following statements about linked list delete is FALSE.
Yang manakah di antara berikut adalah penyataan PALSU mengenai pembuangan senarai berpaut.
   A. Deletion of a node from the middle of the list requires that its predecessor to be changed.
   Membuang satu nod di tengah senarai memerlukan node sebelumnya ditukar.
   B. Deletion of the first node requires that the head pointer to be changed.
   Membuang nod pertama memerlukan kepala penunjuk ditukar.
   C. Deletion of the last node requires a separate test to set the predecessor's link node to 0.
   Membuang nod terakhir memerlukan pengujian yang berasingan untuk mensetkan pautan node sebelum kepada 0.
   D. The deleted node must be recycled.
   Nod yang dibuang mesti diguna semula.

3. A linear list where all addition and deletion are restricted to only one end refers to
Satu senarai berpaut di mana kesemua kemasukan dan pembuangan adalah terhad hanya pada satu hujung merujuk kepada
   A. Linked List / *Senarai berpaut*
   B. Stack / *Timbunan*
   C. Queue / *Giliran*
   D. Tree / *Pepohon*

4. Choose correct information about 'Create List' algorithm.
Pilih maklumat yang benar mengenai algoritma 'Membina Senarai'.
   i. At the end of the process, head node is allocated to the list.
   Pada akhir proses, nod kepala dimasukkan ke dalam senarai.
   ii. A newly created list is actually an empty list.
   Senarai yang baru dibina ialah suatu senarai kosong.
   iii. There is no node in the list, so the count is set to zero.
   Tidak terdapat sebarang nod dalam senarai, maka medan 'count' diset kepada sifar.
   iv. The head node pointer is set to null.
   Penunjuk nod kepala diset kepada 'tiada nilai'.

   A. i, ii, iii
   B. i, ii, iv
   C. i, iii, iv
   D. ii, iii, iv

5. Choose the TRUE statement from the followings:
Pilih penyataan yang BENAR daripada yang berikut:
   i. A header node in linked list is physically positioned so that it is always the first node in the linked list.
   Nod kepala dalam senarai berpaut secara fizikalnya diposisikan supaya ia sentiasa menjadi nod pertama dalam senarai berpaut.
   ii. Stack count function returns the number of elements currently in the stack.
   Fungsi 'Stack count' mengembalikan bilangan elemen terkini di dalam timbunan.
   iii. A pseudocode statement pNew->count = 0 means that the linked-list is empty.
   Penyataan pseudokod pNew->count = 0 bermaksud senarai berpaut adalah kosong.

   A. i , ii , iii
   B. i , ii
   C. ii , iii
   D. i , iii

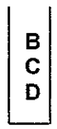
6. Two "start pointer", first node and last node. Forward pointer of the last node points to the first node and backward pointer of the first node points to the last node. Both statements refer to
Dua "penunjuk mula", nod pertama dan nod terakhir. Penunjuk kehadapan nod terakhir menunjuk kepada nod pertama dan penunjuk kebelakang nod pertama menunjuk kepada nod terakhir. Kedua-dua penyataan tersebut merujuk kepada
   A. Singly linked list / *Senarai berpaut 'singly'*
   B. Circular singly linked list / *Senarai berpaut 'singly' berpusing*
   C. Doubly linked list / *Senarai berpaut 'doubly'*
   D. Circular doubly linked list / *Senarai berpaut 'doubly' berpusing*

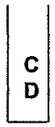
7. *Choose right answer for postfix statement below.*
Pilih jawapan yang betul untuk ungkapan posfik dibawah.

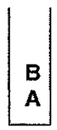
40 24 6 + 10 - /

   A. 0.5
   B. 1
   C. 2
   D. -2

8. If you **push** the letters A, B, C and D in order into a stack of character and then **pop** two times before calling **stack top** operation. What will the stack look like at the end of the process?
*Jika anda **push** abjad A, B, C dan D mengikut turutan ke dalam satu timbunan aksara dan kemudian **pop** dua kali sebelum memanggil operasi **stack top.** Apakah timbunan yang akan kelihatan di akhir proses?*

A.


C.


B.


D.


9. Which algorithm could effectively solve the problem of 'Towers of Hanoi'?
Algoritma manakah boleh menyelesaikan masalah 'Towers of Hanoi' dengan berkesan?

A. Doubly-linked list / *Senarai berpaut 'doubly'*
B. Stack and Queue / *Timbunan dan Baris*
C. Sorting / *Isihan*
D. Recursive / *Ulangan*

10. Choose the correct pretfix statement for infix statement below.
Pilih penyataan prefik yang betul untuk penyataan infik di bawah.

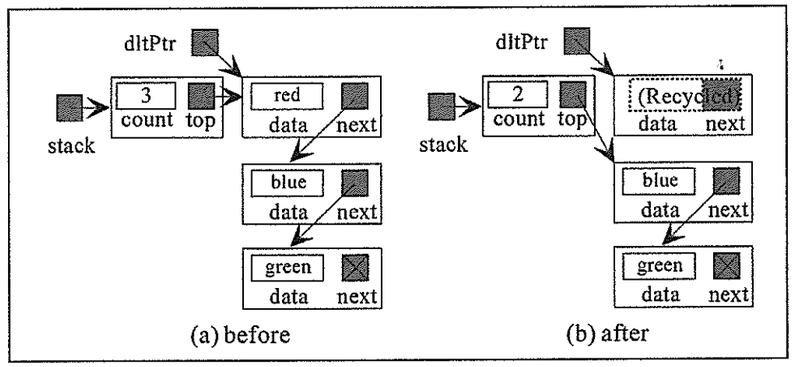
P - (R + S * N) / F

A. - / + * P R S N F
B. - P / + R * S N F
C. P R S N * + F / -
D. - / +* R S N P F

11. Which operation can insert an element after the head node of the data structure?
Operasi manakah boleh menambah elemen selepas nod kepala struktur data?

A. Linear list and Stack / *Senarai linear dan Timbunan*
B. Stack and Queue / *Timbunan dan Baris*
C. Linear list and Queue / *Senarai Linear dan Baris*
D. Stack and Tree / *Timbunan dan Pokok*

12. Given a diagram of a logical operation in processing stack structure. Study the diagram carefully and choose which option defines true about the diagram.
Diberi rajah yang menunjukkan operasi logikal dalam memproses struktur data timbunan. Analisa rajah dan pilih yang manakah benar menerangkan rajah ini.



A. Empty stack
B. Stack top
C. Push stack
D. Pop stack

13. Which definition defines best about the **level** of a node in a tree structure?
*Definisi manakah yang tepat menerangkan mengenai **paras** suatu nod dalam struktur pepohon?*

A. The level of the node is the distance of the leaf in the longest path from the root.
Paras suatu nod adalah jarak antara daun pada laluan terpanjang dari akar.
B. The level of the node is the distance of the node from the root.
Paras suatu nod adalah jarak antara nod dan akar.
C. The level of the node is the distance of the leaf from the root plus one.
Paras suatu nod adalah jarak antara daun dan akar ditambah satu.
D. The level of the node is a sequence of nodes in which each node is adjacent to the next one.
Paras suatu nod adalah turutan nod-nod di mana setiap nod adalah bersebelahan dengan nod selepasnya.

14. Which function will be call to examines the element at the front of the queue?
Fungsi manakah yang akan dipanggil untuk memeriksa elemen pada hadapan baris gilir?

A. Enqueue
B. Dequeue
C. Queue rear
D. Queue front

15. Listing below shown type linear data structure EXCEPT_____ .
Senarai di bawah menunjukkan jenis struktur data linear KECUALI _____ .

A. Linear Linked List / *Senarai* Berpaut Linear
B. Graph / *Graf*
C. Queue / *Baris Gilir*
D. Stack / *Timbunan*

ANSWER SECTION A AND B / *RUANGAN JAWAPAN BAHAGIAN A DAN B*

Answers for Section A **[10M]**
Jawapan untuk Bahagian A:

Question / *Soalan*	Answer / *Jawapan*
Example/*Contoh*:	TRUE
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	

Answers for Section B **[30M]**
Jawapan untuk Bahagian B:

Question / *Soalan*	Answer / *Jawapan*
Example/*Contoh*:	A
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	

SECTION C: STRUCTURE [48 MARKS]
BAHAGIAN C: STRUKTUR [48 MARKAH]

QUESTION / *SOALAN 1*

a) Give ONE (1) reason why data structure is important to a programmer? **[1M]**
Berikan SATU (1) sebab mengapa struktur data adalah penting kepada seorang pengaturcara?
Answer / *Jawapan:*

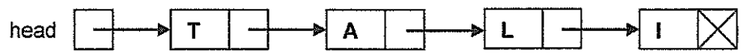
b) What are the differences between data structure and algorithm? **[2M]**
Apakah perbezaan diantara struktur data dan algoritma?
Answer / *Jawapan:*

c) Give ONE (1) example how data structure being apply in real life problem. **[2M]**
Beri SATU (1) contoh bagaimana struktur data diaplikasikan dalam masalah kehidupan sebenar.
Answer / *Jawapan:*

QUESTION / *SOALAN 2*

a) What does the following function do for a given following linked list with first node as head? **[2M]**
Apa yang dibuat oleh fungsi berikut untuk satu senarai berpaut berikut yang diberi dengan node pertama sebagai kepala?

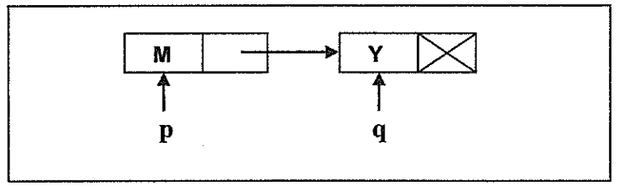
```
void fun1(node* head)
{
    if(head == NULL)
      return;
    fun1(head->next);
    cout << head->data << endl;
}
```



Answer / *Jawapan:*

b) Imagine that you have 2 variables pointer, p and q and also 2 nodes with data field and next fields. Based on the following diagram, give result output after statement **cout** being execute. (Each statement is not related). **[3M]**

*Andaikan anda mempunyai 2 pembolehubah penunding, p dan q serta 2 nod yang mempunyai medan data dan medan nod berikut. Berdasarkan kepada rajah berikut, berikan hasil keluaran setelah penyataan **cout** dilaksanakan (Setiap penyataan adalah tidak berkaitan antara satu sama lain).*



i. p = p -> next; cout << p->data;

Answer / *Jawapan:*

ii. q = p->next; cout << q->data;

Answer / *Jawapan:*

iii. q = p; cout << q->data

Answer / *Jawapan:*

QUESTION / *SOALAN* 3

a) Draw the content of Q1, Q2 and S after the command is executed. **[3M]**

Lukis kandungan Q1, Q2 dan S selepas arahan tersebut dilaksanakan.

Answer / *Jawapan:*

```
Q1=createQueue
Q2=createQueue
S=createStack
enqueue(Q1, 5)
enqueue(Q1, 3)
enqueue(Q2, 7)
enqueue(Q2, 4)
queueRear(Q1, X)
queueFront(Q2, Y)
pushStack(S, X)
pushStack(S, Y)
loop (not emptyQueue Q2)
    dequeue(Q2, Z)
    Z=Z+X
    pushStack(S, Z)
endLoop
popStack(S, X)
enqueue(Q2, topStack(S, Y))
```


b) Change the following infix expression to postfix expression using the algorithmic method (a stack). **[5M]**

Tukarkan ungkapan infix berikut kepada ungkapan postfix dengan menggunakan kaedah algoritma (satu tindanan).

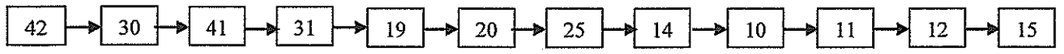
Answer / *Jawapan:*

infix expression *ungkapan infix*	Stack *Tindanan*	postfix expression *ungkapan postfix*
m / (n + (x – s + y) * z) - t		EMPTY

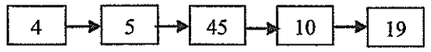
c) Contents of queue Q1 and queue Q2 are as shown below. What would be the content of queue Q3 after the following code is executed? The queue contents are shown front (left) to rear (rignt). **[1M]**

Kandungan giliran Q1 dan giliran Q2 ditunjukkan seperti di bawah. Apakah kandungan giliran Q3 selepas code berikut dilaksanakan? Kandungan giliran menunjukkan depan (kiri) ke belakang (kanan).

Q1 :



Q2 :



Answer / *Jawapan:*

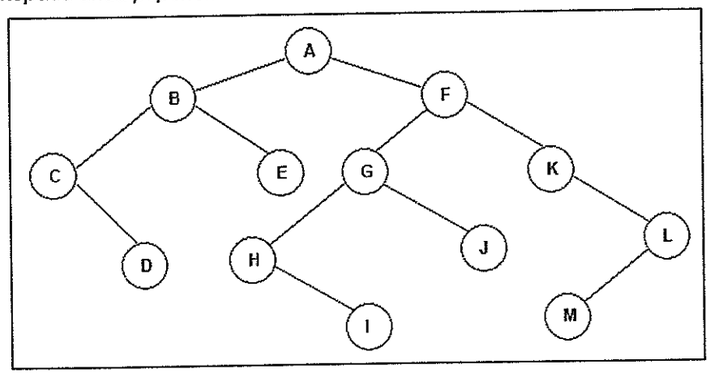
Q3 :

```
1   Q3 = createQueue
2   count = 0
3   loop (not empty Q1 and not empty Q2)
    1   count = count + 1
    2   dequeue (Q1, x)
    3   dequeue (Q2, y)
    4   if (y < x)
        1   enqueue (Q3, x+count)
    5   end if
4   end loop
```

QUESTION / *SOALAN 4*

a) Given to you the following binary tree. Write down the following traversal. **[8M]**

Diberikan kepada anda pepohon binari berikut. Tuliskan rentasan-rentasan berikut.



i. Inorder Traversal

Answer / *Jawapan:*

ii. Postorder Traversal

Answer / *Jawapan:*

iii. Preorder Traversal

Answer / *Jawapan:*

iv. Breadth First Traversal

Answer / *Jawapan:*

b) Refer to binary tree above (a), identify the nodes below. **[2M]**

Rujuk pepohon di atas (a), kenalpasti nod-nod di bawah.

i. internal nodes / *nod-nod dalaman*

Answer / *Jawapan:*

ii. leaf nodes / *nod-nod daun*

Answer / *Jawapan:*

c) Draw expression tree for expression below. **[2M]**

Lukiskan pepohon ungkapan untuk ungkapan di bawah.

A / (B – C + D) * (E – F)

Answer / *Jawapan:*

d) Show the result of inserting 24, 13, 22, 45, 30, 56, 20, 45, 50 into an initially empty binary search tree. This tree will facilitate duplicate elimination. **[2M]**

Tunjukkan hasil kemasukan 24, 13, 22, 45, 30, 56, 20, 45, 50 *ke dalam satu pepohon binari yang dinilaiawalkan kosong. Pepohon ini akan memudahkan penghapusan pendua.*

Answer / *Jawapan:*

e) Based on above binary search tree (d). Show the balance factor in the resulting tree. **[1M]**

Berdasarkan pada pohon carian binari di atas (d). Tunjukkan faktor keseimbangan dalam pepohon yang dihasilkan.

Answer / *Jawapan:*

QUESTION / *SOALAN 5*

a) Consider the following algorithm.

What would be returned if recFun is called in each statement below. **[2M]**

Pertimbangkan algoritma berikut.

Apakah yang akan dipulangkan jika recFun dipanggil dalam setiap penyataan di bawah.

```
Algorithm recFun ( x <integer>, y  <integer>)
1  If (x < y)
   1   return -3
2   else
   1   return (recFun( x-y,  y+3 )+y)
3   end if
```

i. recFun(2, 7)

Answer / *Jawapan:*

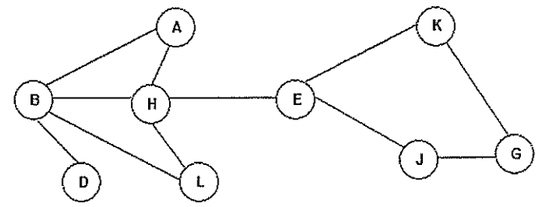
ii. recFun(15, 3)

Answer / *Jawapan:*

b) Give **depth first traversal** for the graph below. Please start from E. **[3M]**

*Berikan **penjelajahan dalaman pertama** untuk graf di bawah. Sila mula dari E.*

Answer / *Jawapan:*



QUESTION / *SOALAN 6*

a) An array contains the elements shows below. Choose between sequential or binary search algorithm to find 56. Show the tracing steps in each loop iteration. **[3M]**

Satu array mengandungi elemen-elemen dipaparkan di bawah. Pilih antara algoritma carian jujukan atau binari untuk mencari 56. Tunjukkan jejak langkah yang diikut dalam setiap ulangan lelaran.

8 13 17 30 37 44 56 63 69 71 88 97

Answer / *Jawapan:*

b) Given a list of numbers: *24 18 55 2 94 31 45*. Sort the list (by drawing a sort diagram for each phase) by using bubble sort. **[4M]**

Diberi suatu senarai nombor: 24 18 55 2 94 31 45. Isih senarai ini (dengan melukiskan rajah isihan untuk setiap fasa) dengan menggunakan isihan buih.

Answer / *Jawapan:*

ORIGINAL: 24 18 55 2 94 31 45

c) What is the time complexity of the following program. **[2M]**

Apakah kerumitan masa program berikut.

Answer / *Jawapan:*

```
void fun(int n){
   int i,j, count=0;
   for (i=0; i<n; i++)
     for (j=1; j<n; j=j*2)
         count=count+n;
}
```


SECTION D: PROGRAMMING AND ALGORITHM [12 MARKS]

BAHAGIAN D: PENGATURCARAAN DAN ALGORITMA [12 MARKAH]

Given to you declaration class structure in STACK.h and QUEUE.h below. Write testApplication.cpp to solve question below using combination stack and queue structure. There will one queue object and two stack object.

Read a sentence, enqueue each of the character into a queue. Each time reading the character check it whether it is consonant or vowel. Push consonant character into stack consonant and push vowel character into vowel stack. Inside Queue there were only consonant character leave inside. Example of the output shown as below.

Diberikan kepada anda struktur pengisytiharan kelas dalam STACK.h dan QUEUE.h di bawah. Tulis testApplication.cpp untuk menyelesaikan soalan di bawah menggunakan struktur timbunan dan giliran. Akan ada satu objek giliran dan dua objek timbunan

Baca satu ayat, masukkan setiap aksara ke dalam giliran. Setiap kali membaca aksara, periksa sama ada ia konsonan atau vokal. Masukkan aksara konsonan ke dalam timbunan konsonan dan masukkan aksara vokal ke dalam timbunan vokal. Di dalam gilirian hanya ada watak konsonan yang tersisa di dalamnya. Contoh output ditunjukkan seperti di bawah.

STACK.h

```
class STACK{

   private:
      NODE *top;
      int count;

   public:
     STACK();
     bool stackEmpty();
     bool push(DATA dataIn);
     bool pop(DATA &dataOut);
     bool stackTop(DATA &dataOut);
};
```

QUEUE.h

```
class QUEUE{

   private:
      NODE *front;
      NODE *rear;
      int count;

   public:
     QUEUE();
     bool stackEmpty();
     bool enqueue(DATA dataIn);
     bool dequeue(DATA &dataOut);
     bool queueFront(DATA &dataOut);
     bool queueRear(DATA &dataOut);
};
```

Insert a sentence: **Congratulation You Win**

Display Queue: **CngrtltnYWn**

Display Vowel Stack: **iuooiauao**

Display Consonant Stack: **nWYntltrgnC**