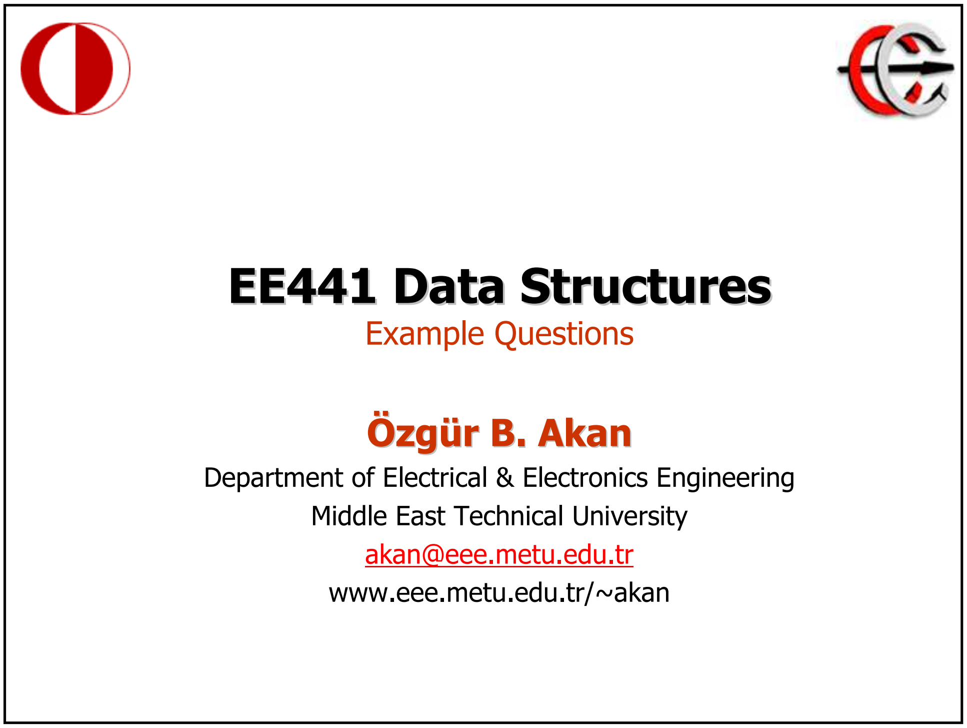
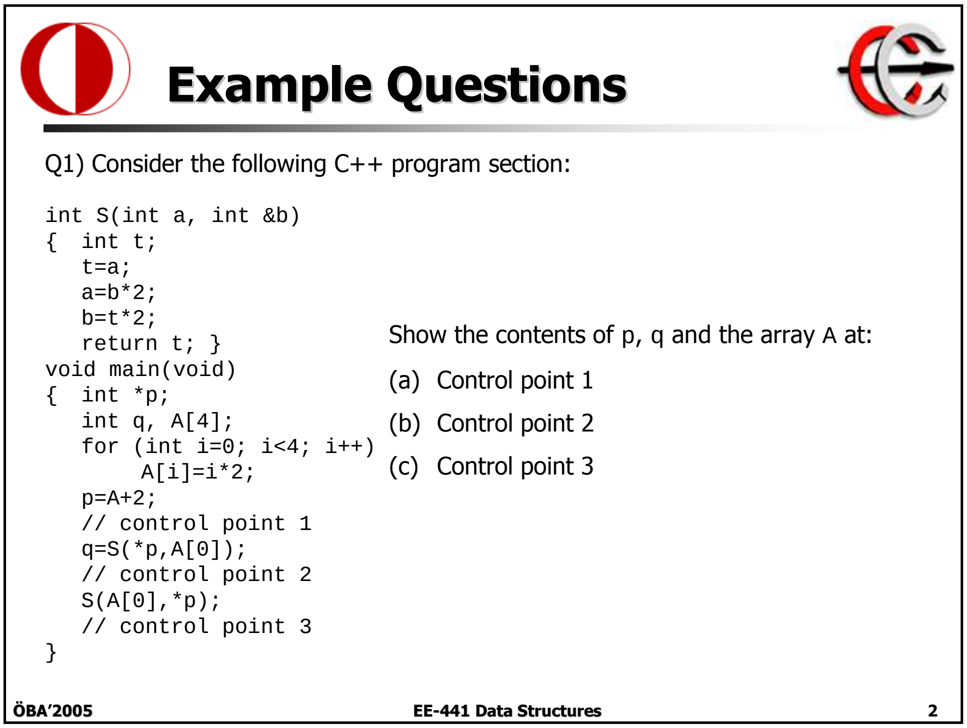


EE441 Data Structures

Example Questions

Özgür B. Akan

Department of Electrical & Electronics Engineering
Middle East Technical University
akan@eee.metu.edu.tr
www.eee.metu.edu.tr/~akan



Example Questions

Q1) Consider the following C++ program section:

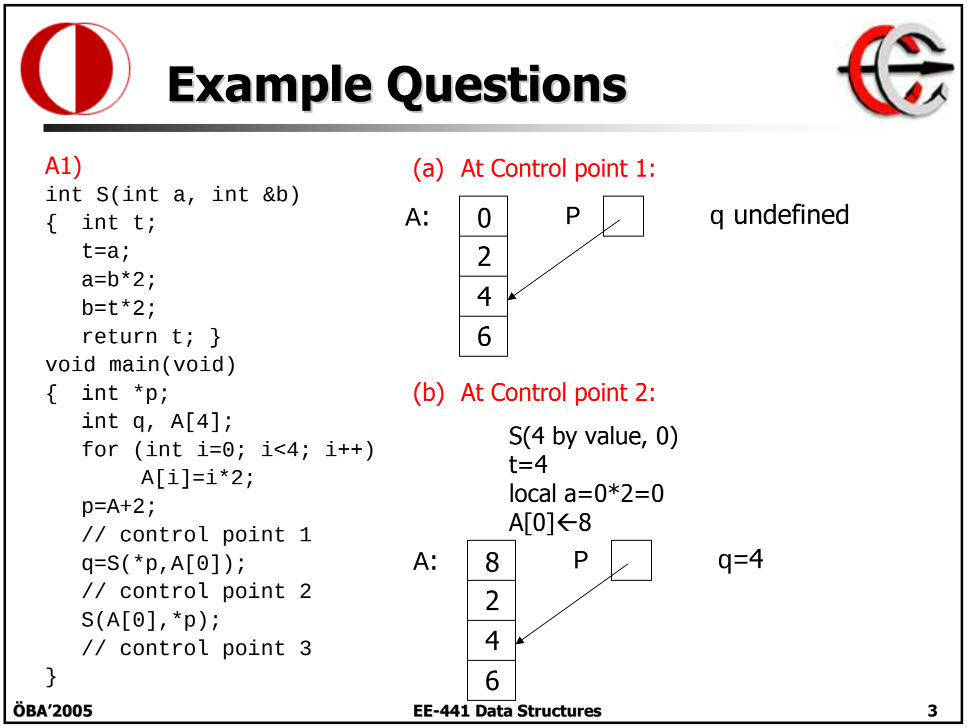
```
int S(int a, int &b)
{  int t;
   t=a;
   a=b*2;
   b=t*2;
   return t; }
void main(void)
{  int *p;
   int q, A[4];
   for (int i=0; i<4; i++)
        A[i]=i*2;
   p=A+2;
   // control point 1
   q=S(*p,A[0]);
   // control point 2
   S(A[0],*p);
   // control point 3
}
```

Show the contents of p, q and the array A at:

(a) Control point 1

(b) Control point 2

(c) Control point 3



Example Questions

A1)

```
int S(int a, int &b)
{  int t;
   t=a;
   a=b*2;
   b=t*2;
   return t; }
void main(void)
{  int *p;
   int q, A[4];
   for (int i=0; i<4; i++)
        A[i]=i*2;
   p=A+2;
   // control point 1
   q=S(*p,A[0]);
   // control point 2
   S(A[0],*p);
   // control point 3
}
```

(a) At Control point 1:

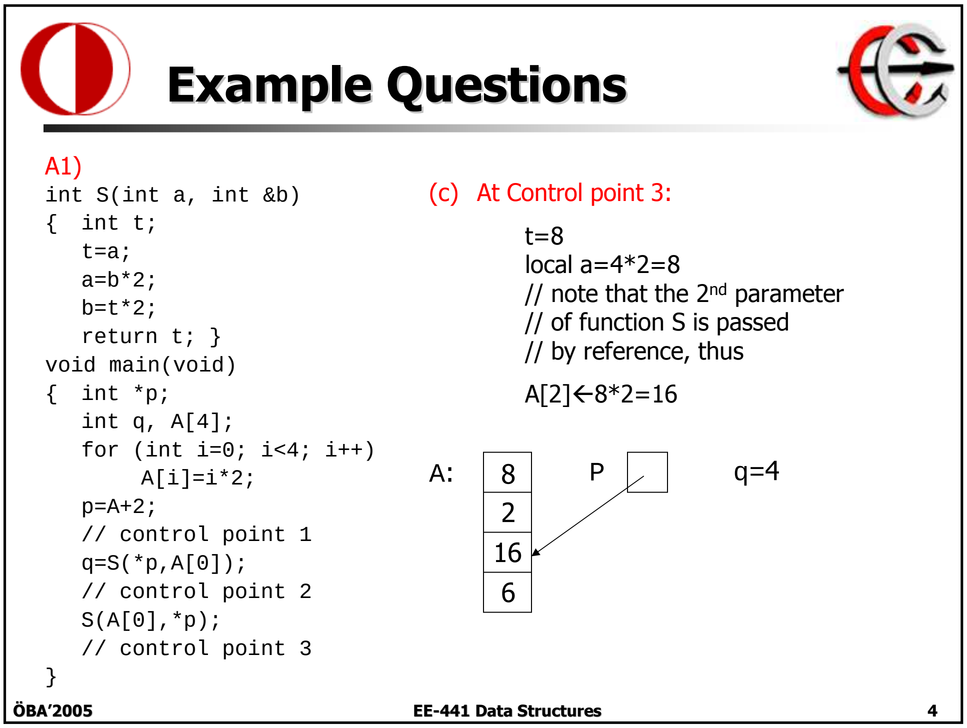
A: 0 2 4 6 P q undefined

(b) At Control point 2:

S(4 by value, 0)
t=4
local a=0*2=0
A[0]←8

A: 8 2 4 6 P q=4

ÖBA'2005 EE-441 Data Structures 3



Example Questions

A1)

```
int S(int a, int &b)
{  int t;
   t=a;
   a=b*2;
   b=t*2;
   return t; }
void main(void)
{  int *p;
   int q, A[4];
   for (int i=0; i<4; i++)
        A[i]=i*2;
   p=A+2;
   // control point 1
   q=S(*p,A[0]);
   // control point 2
   S(A[0],*p);
   // control point 3
}
```

(c) At Control point 3:

t=8
local a=4*2=8
// note that the 2nd parameter
// of function S is passed
// by reference, thus

A[2]←8*2=16

A: 8 2 16 6 P q=4

ÖBA'2005 EE-441 Data Structures 4

Example Questions

Q2)

(a) What is the order (time complexity in O(.) notation) of the following tasks in the worst case?

i. computing the sum of the first n even integers by using a `for` loop

ii. pushing an item to a stack of n items

(b) Order the following functions by growth rate: *$n\log n$, $n\log\log n$, $\log^2 n$, $n\log n^2$, $2^{n/2}$, 2^n, 37, $n^2\log n$, 2000*. Indicate the group of functions which grow at the same rate.

(c) Suppose that the implementation of a particular algorithm appears in C++ as follows:

```
for (int i=1; i>=n; i++)
{ for (int j=1; j>=i; j++)
  {     for (int k=1; k>=10; k++)
                myfunction(i,j,k,n);
  }
}
```

What is the time complexity of the algorithm, assuming that `myfunction` has a running time, which is O(log n)? Justify your answer.

Example Questions

A2)

(a) i. computing the sum of the first n even integers by using a `for` loop

```
        for (i=1;i<=n;i++) sum=sum+i*2;
```

the work (time it takes…) grows linearly with n, hence O(n)

ii. pushing an item to a stack of n items

Recall the implementation of `Push` method of `Stack` class

```
void Stack::Push(const DataType& item)
{
  if (top==MaxStackSize-1)
  {
        cerr<<"Stack overflow"<<endl; exit(1);
  }
  top++;
  stacklist[top] =item;
}
```

work done in pushing an item to a stack is independent of stack size…Hence, O(1)

Example Questions

A2)

(b)

37, 2000

$\log^2 n$

$n\log\log n$

$n\log n$, $n\log n^2$

$n^2\log n$

$2^{n/2}$, 2^n

```
    for (int i=1; i>=n; i++)
    {    for (int j=1; j>=i; j++)
         {        for (int k=1; k>=10; k++)
                       myfunction(i,j,k,n);
         }
    }
```

(c) If `myfunction` requires `t` time units, the innermost loop on `k` requires `10*t` time units. The loop of `j` requires `10*t*i` time units and the outermost loop on `i` requires:

$$\sum_{i=1}^{n}(10*t*i) = 10*t*(1+2+\ldots+n) = 10*t*n*\frac{(n+1)}{2}$$

$$= O(t*n^2)$$

Since `t` is O(log n) then the overall execution time is $O(n^2\log n)$

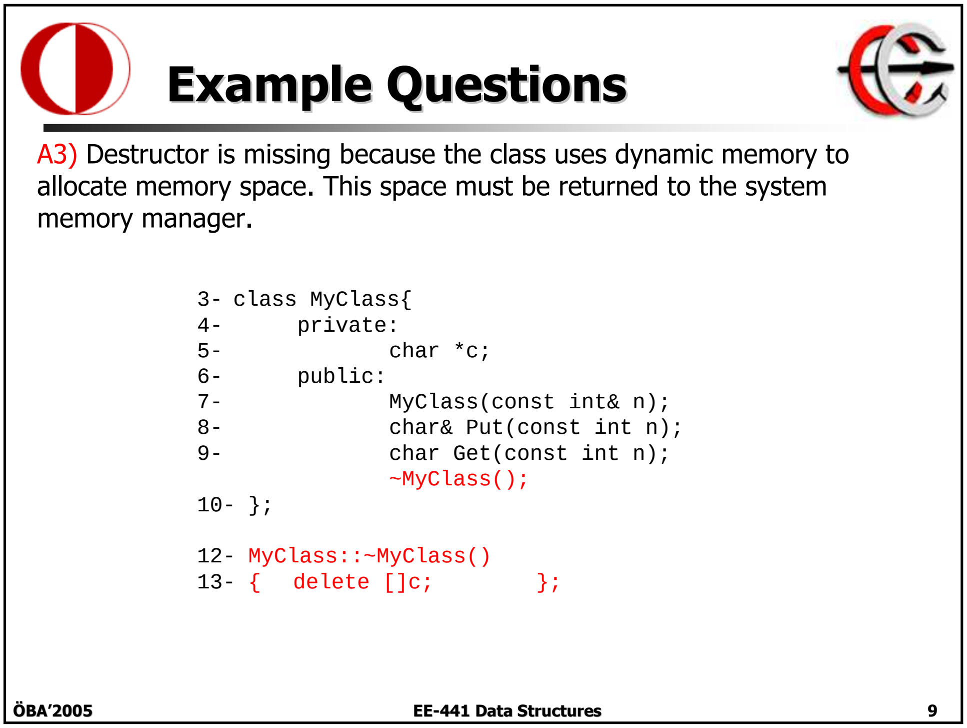
Example Questions

Q3) Consider the following C++ program

```
1-  #include <iostream.h>
2-
3-  class MyClass{
4-       private:
5-        char *c;
6-       public:
7-        MyClass(const int& n);
8-        char& Put(const int n);
9-        char Get(const int n);
10- };
11-
12- MyClass::MyClass(const int& n)
13- {    c= new char[n]       };
14-
15- char& MyClass::Put(const int& n)
16- {    return c[n];         };
17-
18-      char& MyClass::Get(const int& n)
19- {    return c[n];         };
20-
```

```
21- void MyFn(MyClass& m1)
22- {
23-      MyClass *mc;
24-
25-      mc = new MyClass(10);
26-      mc->Put(3) = ‘a’;
27-      m1.Put(3) = mc->Get(3);
28- };
29-
30- main()
31- {
32-      int n;
33-      MyClass m1(20);
34-
35-      cin >> n;
36-      MyFn(m1);
37-      cout << m1.Get(3);
38- };
```

What is the major programming error related to dynamic memory usage in this program? Indicate how you would correct this error by giving the line numbers of the lines you would delete, if necessary, and adding new code, if necessary.

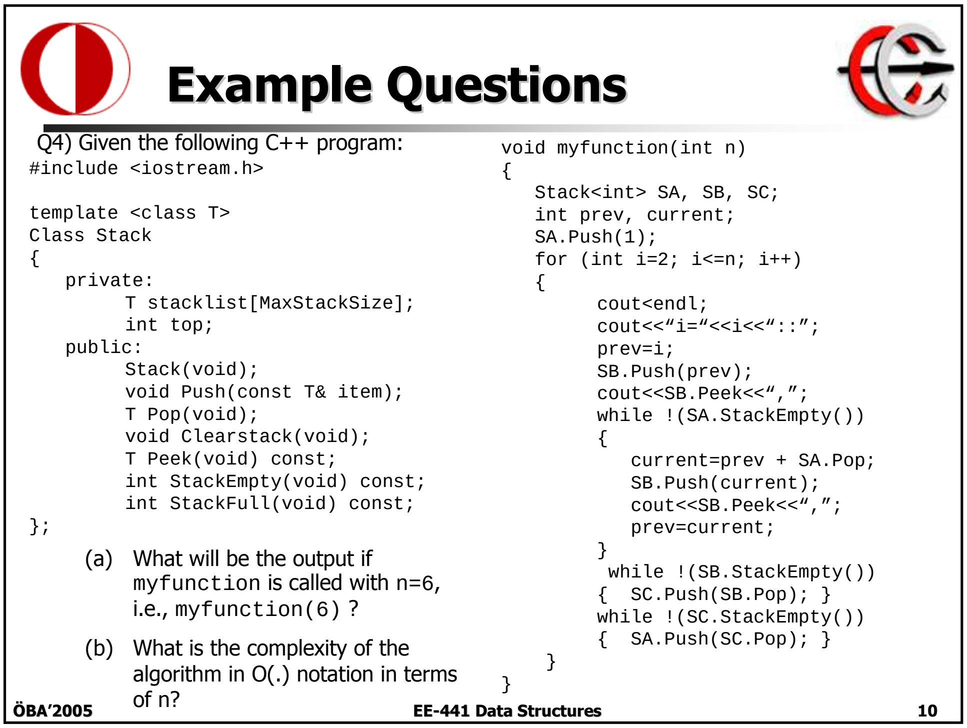


Example Questions

A3) Destructor is missing because the class uses dynamic memory to allocate memory space. This space must be returned to the system memory manager.

```
3- class MyClass{
4-      private:
5-              char *c;
6-      public:
7-              MyClass(const int& n);
8-              char& Put(const int n);
9-              char Get(const int n);
                ~MyClass();
10- };

12- MyClass::~MyClass()
13- {   delete []c;        };
```



Example Questions

Q4) Given the following C++ program:

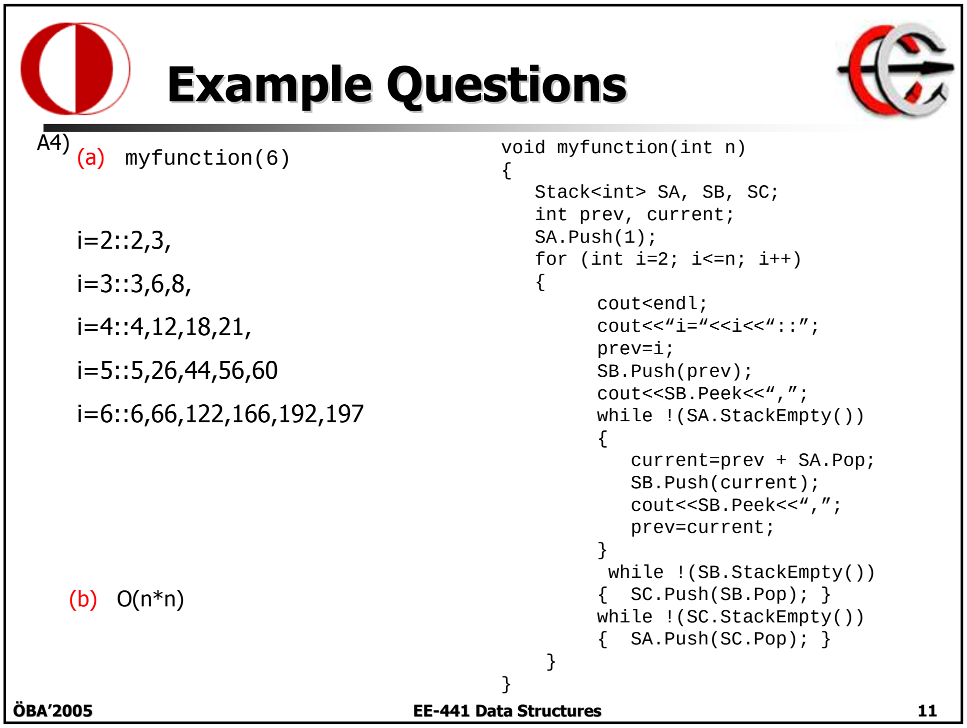
```
#include <iostream.h>

template <class T>
Class Stack
{
   private:
        T stacklist[MaxStackSize];
        int top;
   public:
        Stack(void);
        void Push(const T& item);
        T Pop(void);
        void Clearstack(void);
        T Peek(void) const;
        int StackEmpty(void) const;
        int StackFull(void) const;
};
```

(a) What will be the output if `myfunction` is called with n=6, i.e., `myfunction(6)` ?

(b) What is the complexity of the algorithm in O(.) notation in terms of n?

```
void myfunction(int n)
{
   Stack<int> SA, SB, SC;
   int prev, current;
   SA.Push(1);
   for (int i=2; i<=n; i++)
   {
        cout<endl;
        cout<<“i=“<<i<<“::”;
        prev=i;
        SB.Push(prev);
        cout<<SB.Peek<<“,”;
        while !(SA.StackEmpty())
        {
           current=prev + SA.Pop;
           SB.Push(current);
           cout<<SB.Peek<<“,”;
           prev=current;
        }
         while !(SB.StackEmpty())
        {  SC.Push(SB.Pop); }
        while !(SC.StackEmpty())
        {  SA.Push(SC.Pop); }
    }
}
```



Example Questions

A4) (a) myfunction(6)

i=2::2,3,

i=3::3,6,8,

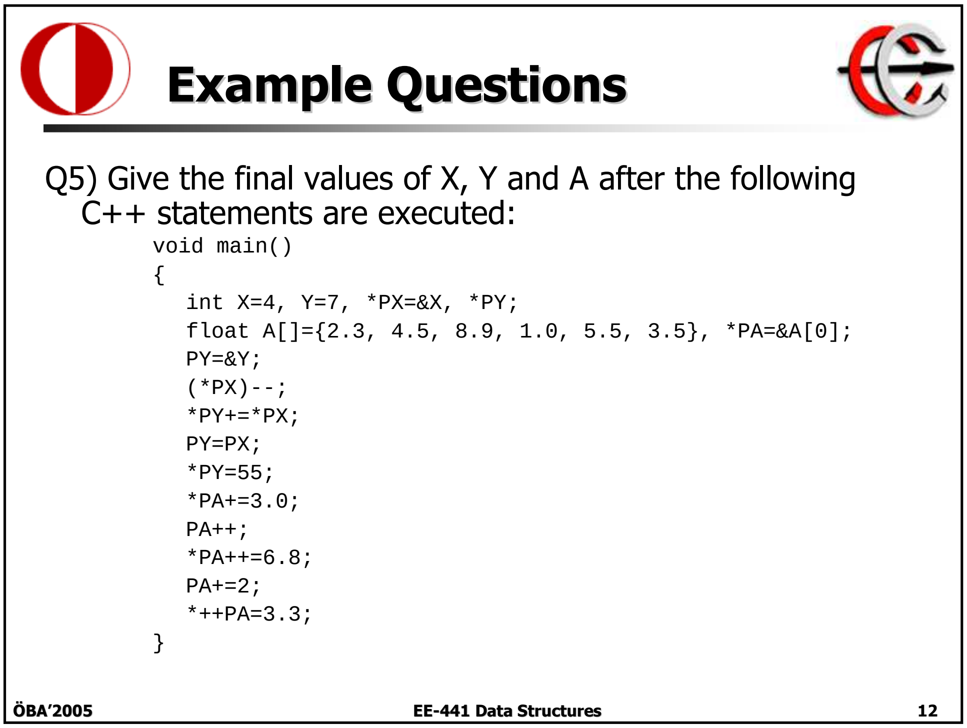
i=4::4,12,18,21,

i=5::5,26,44,56,60

i=6::6,66,122,166,192,197

(b) O(n*n)

```
void myfunction(int n)
{
   Stack<int> SA, SB, SC;
   int prev, current;
   SA.Push(1);
   for (int i=2; i<=n; i++)
   {
        cout<endl;
        cout<<“i=“<<i<<“::”;
        prev=i;
        SB.Push(prev);
        cout<<SB.Peek<<“,”;
        while !(SA.StackEmpty())
        {
           current=prev + SA.Pop;
           SB.Push(current);
           cout<<SB.Peek<<“,”;
           prev=current;
        }
         while !(SB.StackEmpty())
        {  SC.Push(SB.Pop); }
        while !(SC.StackEmpty())
        {  SA.Push(SC.Pop); }
    }
}
```



Example Questions

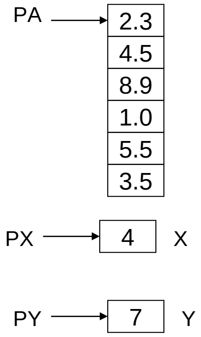
Q5) Give the final values of X, Y and A after the following C++ statements are executed:

```
void main()
{
   int X=4, Y=7, *PX=&X, *PY;
   float A[]={2.3, 4.5, 8.9, 1.0, 5.5, 3.5}, *PA=&A[0];
   PY=&Y;
   (*PX)--;
   *PY+=*PX;
   PY=PX;
   *PY=55;
   *PA+=3.0;
   PA++;
   *PA++=6.8;
   PA+=2;
   *++PA=3.3;
}
```

Example Questions

A5)

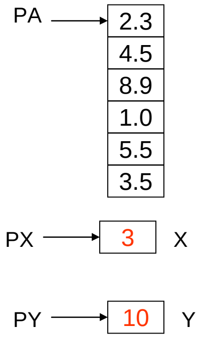
```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```

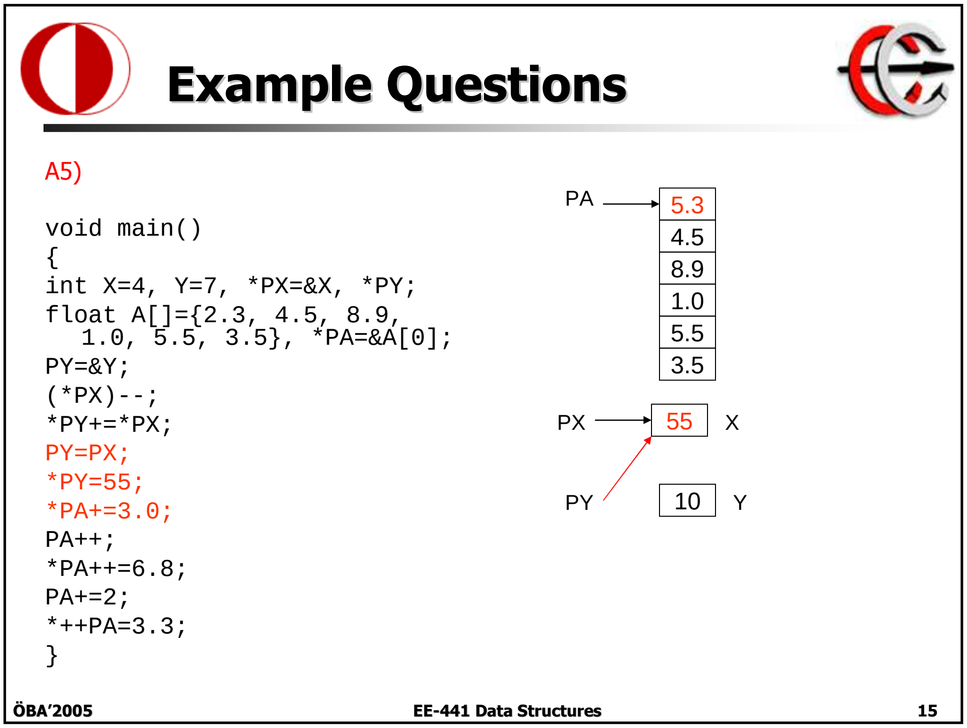


Example Questions

A5)

```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```

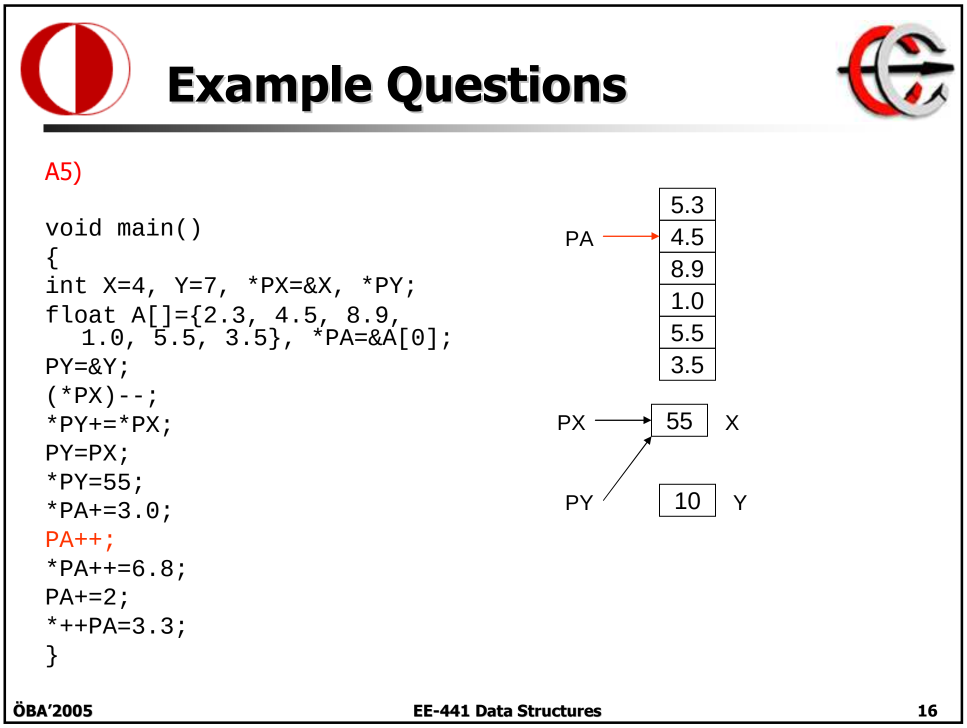




Example Questions

A5)

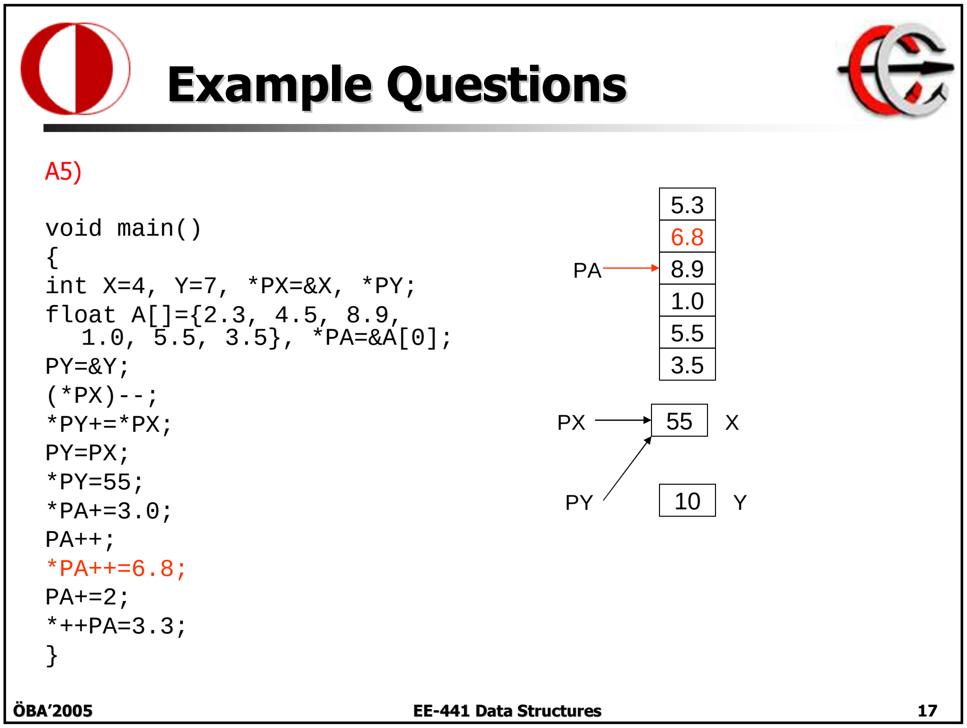
```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```



Example Questions

A5)

```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```



Example Questions

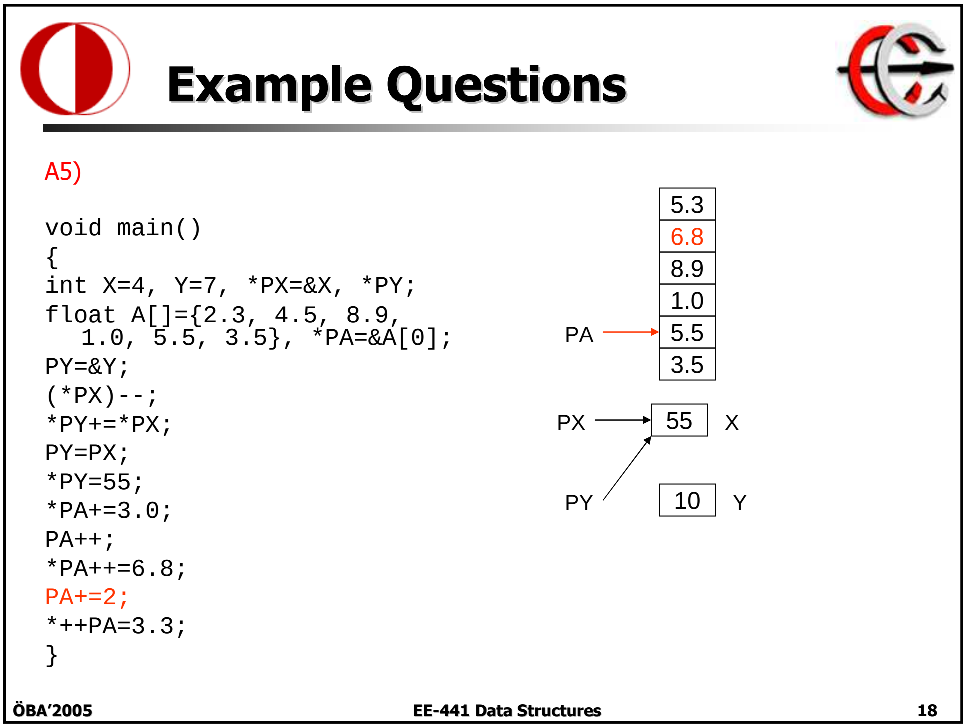
A5)

```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```

5.3
6.8
8.9
1.0
5.5
3.5

PA
PX
55
X
PY
10
Y

ÖBA'2005 EE-441 Data Structures 17



Example Questions

A5)

```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```

5.3
6.8
8.9
1.0
5.5
3.5

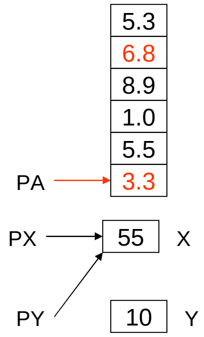
PA
PX
55
X
PY
10
Y

ÖBA'2005 EE-441 Data Structures 18

Example Questions

A5)

```
void main()
{
int X=4, Y=7, *PX=&X, *PY;
float A[]={2.3, 4.5, 8.9,
   1.0, 5.5, 3.5}, *PA=&A[0];
PY=&Y;
(*PX)--;
*PY+=*PX;
PY=PX;
*PY=55;
*PA+=3.0;
PA++;
*PA++=6.8;
PA+=2;
*++PA=3.3;
}
```

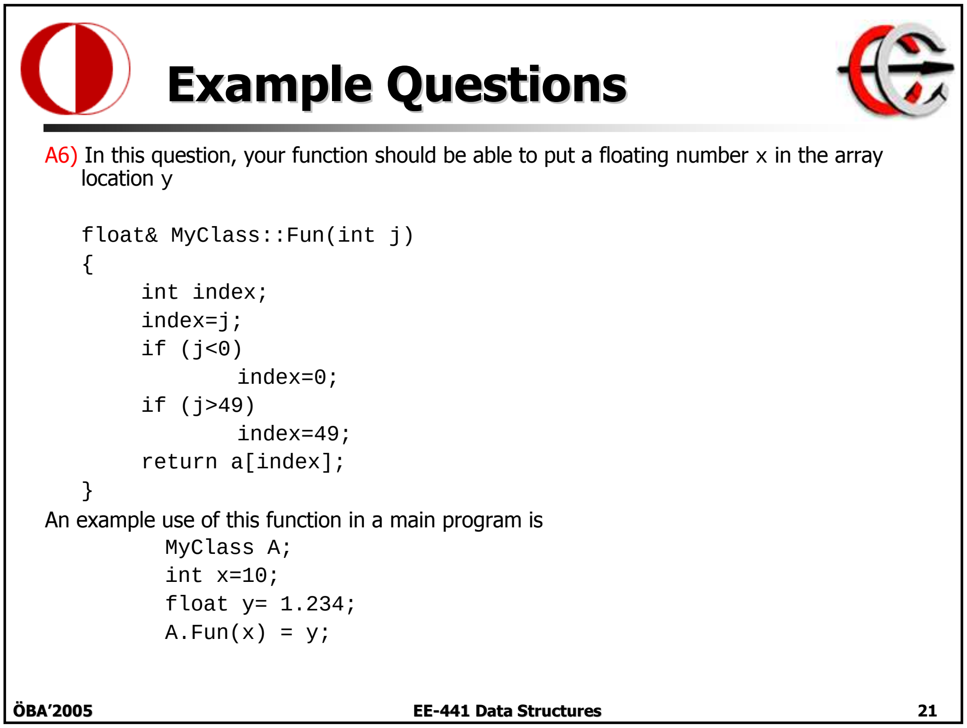


Example Questions

Q6) Given the partial class definition

```
class MyClass
{  private:
      float a[50];
   public:
      MyClass(void);
      ??? Fun(??? J); // single input argument};
```

Implement the member function Fun which will allow the user to insert floating numbers to the desired location in the floating array a. Array indices greater than 49 will be inserted in the last location and negative indices will be inserted in the 1st location in the array. Any index between 0-49 will be inserted in the corresponding location in the array. This member function will take only a SINGLE input argument. Comment on each line of your code



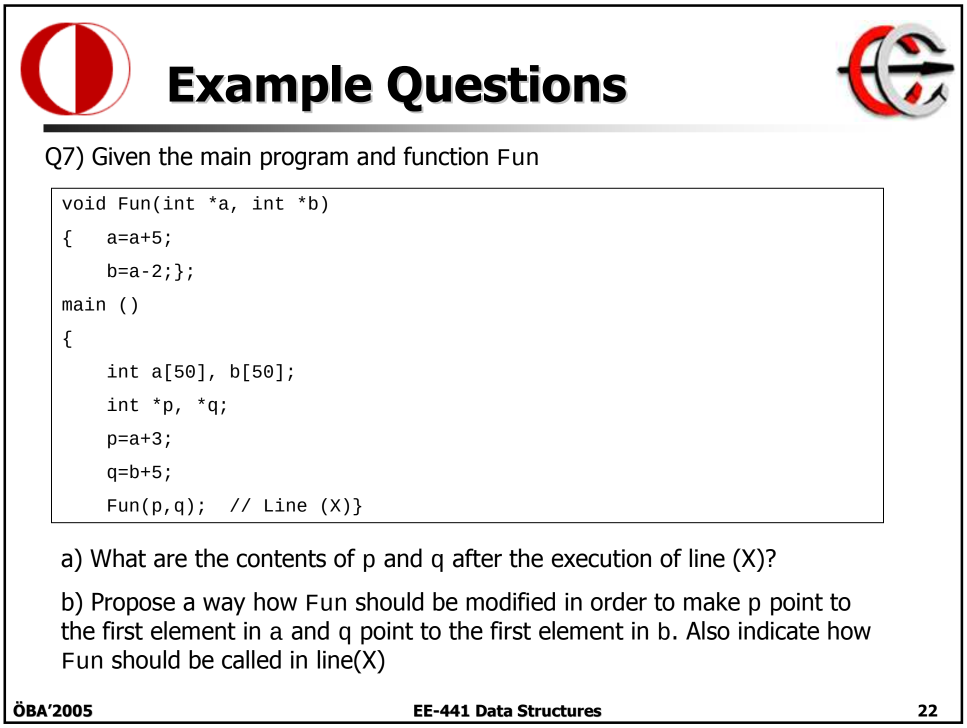
Example Questions

A6) In this question, your function should be able to put a floating number x in the array location y

```
float& MyClass::Fun(int j)
{
     int index;
     index=j;
     if (j<0)
             index=0;
     if (j>49)
             index=49;
     return a[index];
}
```

An example use of this function in a main program is

```
         MyClass A;
         int x=10;
         float y= 1.234;
         A.Fun(x) = y;
```



Example Questions

Q7) Given the main program and function Fun

```
void Fun(int *a, int *b)
{   a=a+5;
    b=a-2;};
main ()
{
    int a[50], b[50];
    int *p, *q;
    p=a+3;
    q=b+5;
    Fun(p,q);  // Line (X)}
```

a) What are the contents of p and q after the execution of line (X)?

b) Propose a way how Fun should be modified in order to make p point to the first element in a and q point to the first element in b. Also indicate how Fun should be called in line(X)

Example Questions

A7)

```
void Fun(int *a, int *b)
{   a=a+5;
    b=a-2;};
main ()
{
    int a[50], b[50];
    int *p, *q;
    p=a+3;
    q=b+5;
    Fun(p,q);  // Line (X)}
```

a) Since the parameters are passed by value, the contents of p and q are unchanged.

Example Questions

A7)

b) Propose a way how Fun should be modified in order to make p point to the first element in a and q point to the first element in b. Also indicate how Fun should be called in line(X)

```
void Fun(int &*a, int &*b)
{   a=a-3;
    b=b-5;};
main ()
{
    int a[50], b[50];
    int *p, *q;
    p=a+3;
    q=b+5;
    Fun(p,q);  // Line (X)}
```

```
void Fun(int **a, int **b)
{   *a=*a-3;
    *b=*b-5;};
main ()
{
    int a[50], b[50];
    int *p, *q;
    p=a+3;
    q=b+5;
    Fun(&p,&q);  // Line (X)}
```

Example Questions

Q8) (a) For the following declaration of the `Stack` class, assuming that the public methods `Stack`, `Push`, `Pop`, `StackEmpty` and `StackFull` are implemented as discussed in class, implement the C++ member function `Count` that will return the number of items in the owner object. Your member function should access the data structure through the existing member functions.

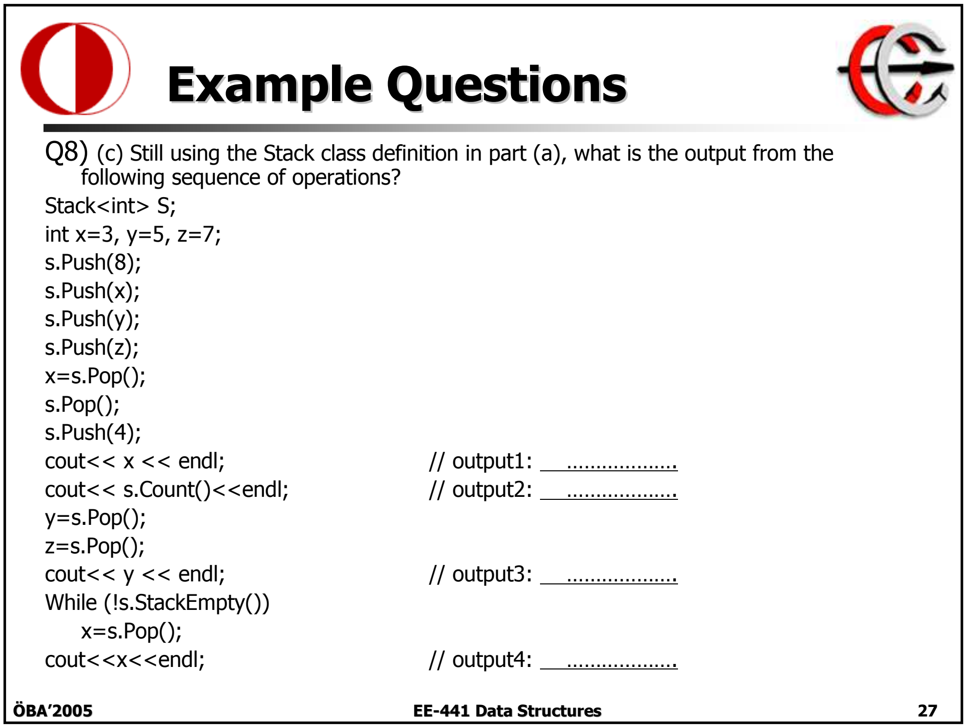
```
template <class T>
class Stack
{private: T stacklist[Maxstacksize];
          int top;
public: Stack(void);
         void Push(const T &item);
         T Pop(void);
         int StackEmpty(void) const;
         int StackFull(void) const;
         int Count(void) const;
```

Example Questions

Q8) (b) Using the `Stack` class definition in part (a), implement a global function

```
template <class T>
int Bottom (Stack<T> &s, T &last)
{ }
```

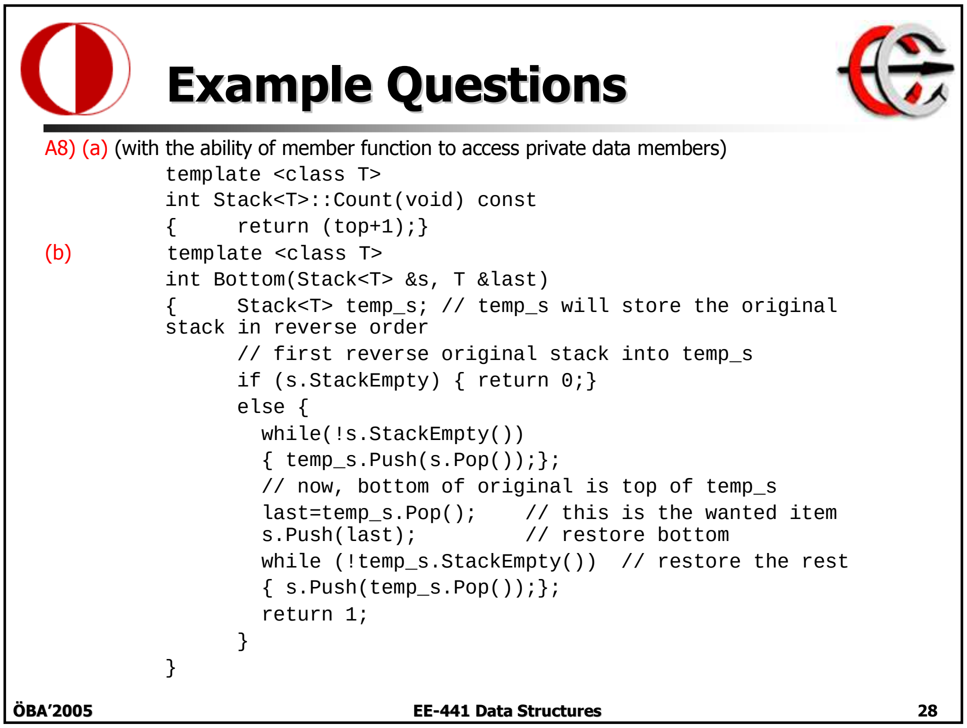
that returns 0 if stack `s` is empty and 1 if it is non-empty, assigning the bottom element to the argument `last`. Upon return from your function the original stack should remain unchanged. Your function implementation should be properly commented for understandability.



Example Questions

Q8) (c) Still using the Stack class definition in part (a), what is the output from the following sequence of operations?

```
Stack<int> S;
int x=3, y=5, z=7;
s.Push(8);
s.Push(x);
s.Push(y);
s.Push(z);
x=s.Pop();
s.Pop();
s.Push(4);
cout<< x << endl;                  // output1: ___.................
cout<< s.Count()<<endl;            // output2: ___.................
y=s.Pop();
z=s.Pop();
cout<< y << endl;                  // output3: ___.................
While (!s.StackEmpty())
    x=s.Pop();
cout<<x<<endl;                     // output4: ___.................
```



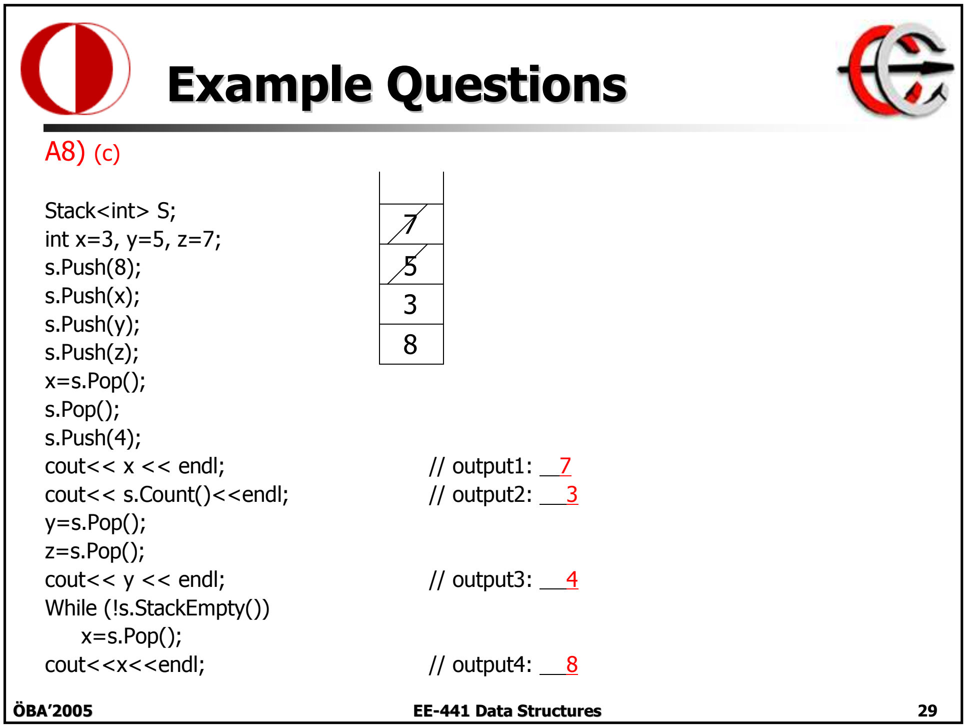
Example Questions

A8) (a) (with the ability of member function to access private data members)

```
          template <class T>
          int Stack<T>::Count(void) const
          {     return (top+1);}
```

(b)

```
          template <class T>
          int Bottom(Stack<T> &s, T &last)
          {     Stack<T> temp_s; // temp_s will store the original
          stack in reverse order
                // first reverse original stack into temp_s
                if (s.StackEmpty) { return 0;}
                else {
                  while(!s.StackEmpty())
                  { temp_s.Push(s.Pop());};
                  // now, bottom of original is top of temp_s
                  last=temp_s.Pop();    // this is the wanted item
                  s.Push(last);         // restore bottom
                  while (!temp_s.StackEmpty())  // restore the rest
                  { s.Push(temp_s.Pop());};
                  return 1;
                }
          }
```

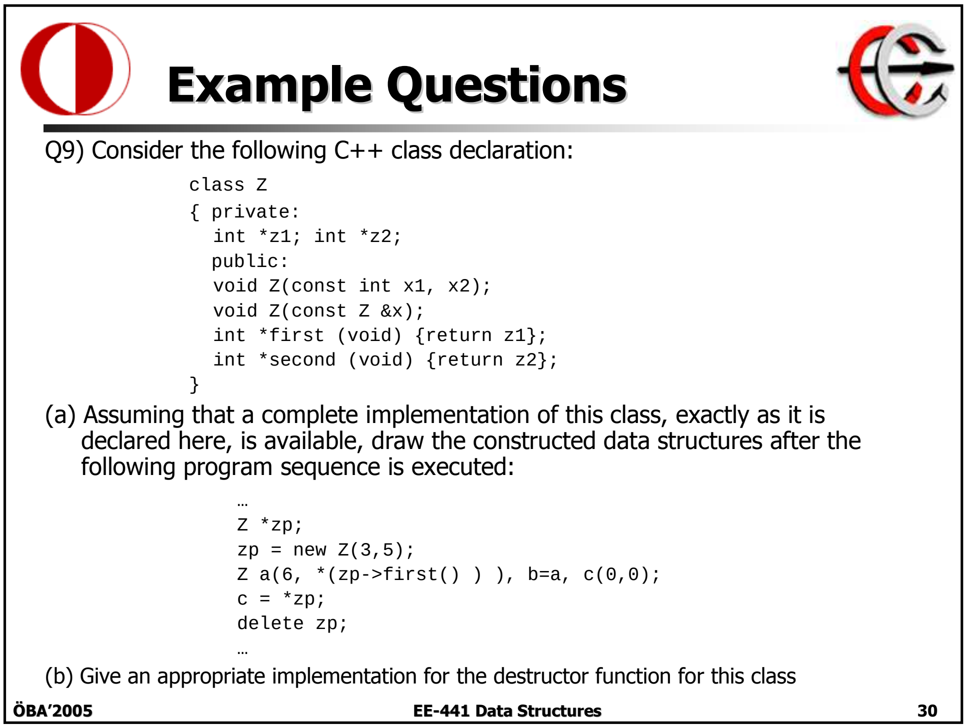


Example Questions

A8) (c)

```
Stack<int> S;
int x=3, y=5, z=7;
s.Push(8);
s.Push(x);
s.Push(y);
s.Push(z);
x=s.Pop();
s.Pop();
s.Push(4);
cout<< x << endl;              // output1: __7
cout<< s.Count()<<endl;        // output2: __3
y=s.Pop();
z=s.Pop();
cout<< y << endl;              // output3: __4
While (!s.StackEmpty())
    x=s.Pop();
cout<<x<<endl;                 // output4: __8
```

~~7~~
~~5~~
3
8



Example Questions

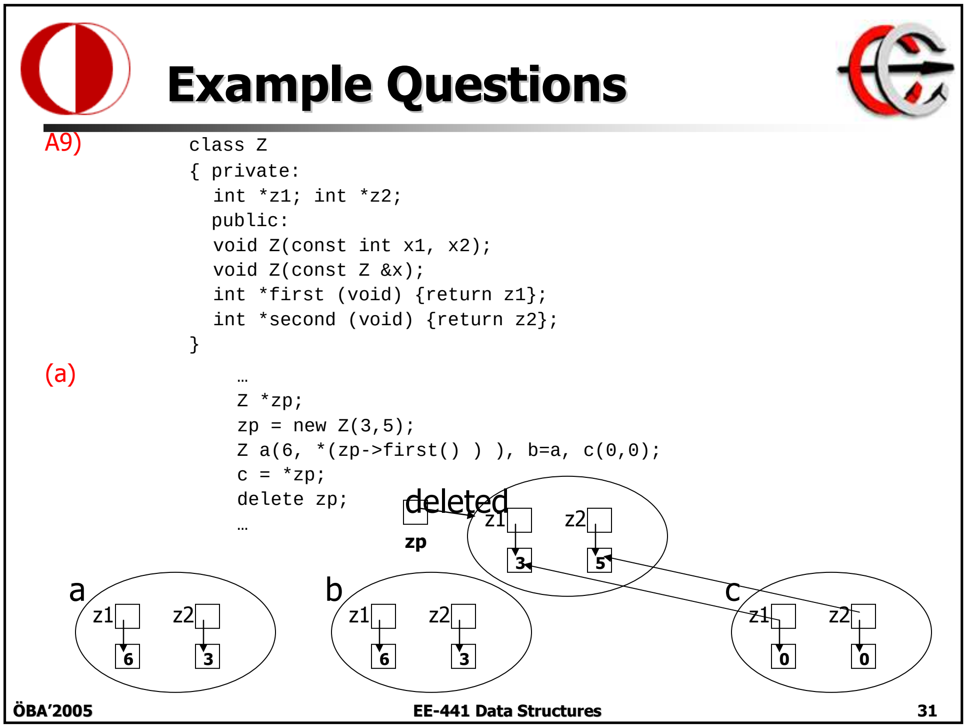
Q9) Consider the following C++ class declaration:

```
class Z
{ private:
    int *z1; int *z2;
  public:
    void Z(const int x1, x2);
    void Z(const Z &x);
    int *first (void) {return z1};
    int *second (void) {return z2};
}
```

(a) Assuming that a complete implementation of this class, exactly as it is declared here, is available, draw the constructed data structures after the following program sequence is executed:

```
…
Z *zp;
zp = new Z(3,5);
Z a(6, *(zp->first() ) ), b=a, c(0,0);
c = *zp;
delete zp;
…
```

(b) Give an appropriate implementation for the destructor function for this class



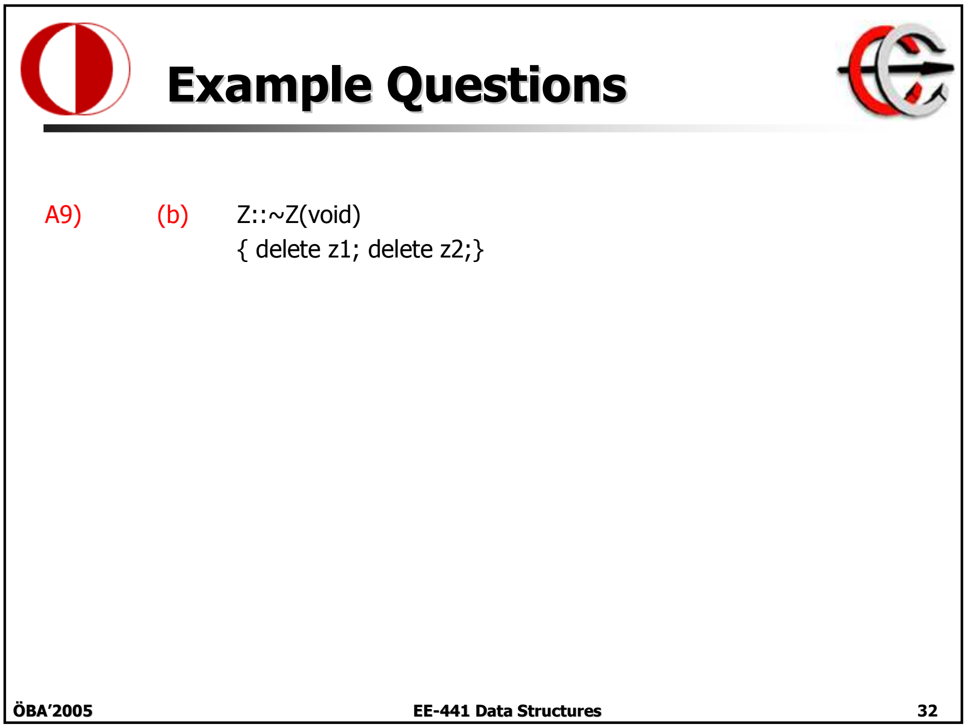
Example Questions

A9)

```
class Z
{ private:
  int *z1; int *z2;
  public:
  void Z(const int x1, x2);
  void Z(const Z &x);
  int *first (void) {return z1};
  int *second (void) {return z2};
}
```

(a)

```
…
Z *zp;
zp = new Z(3,5);
Z a(6, *(zp->first() ) ), b=a, c(0,0);
c = *zp;
delete zp;
…
```



Example Questions

A9) (b) Z::~Z(void)
{ delete z1; delete z2;}

Example Questions

Q10) Consider the following C++ class declaration:

```
template <class T>
class MT2
{ private: int n; T *p;
        public:  MT2(const T &m1, int nn=0)
            { n=nn;
              if (n>0) p= new T[n];
              for (int i=0; i<n; i++) *(p+i) =m1;
            };
            MT2(const MT2<T> &m);
            ~MT2(void);
            MT2<T> &operator= (const MT2<T> &mt2obj);
}
```

a) Draw a diagram that show the data structures created by the following C++ statements:

```
MT2<int> A(1), *q;
q=new MT2(7,5);
```

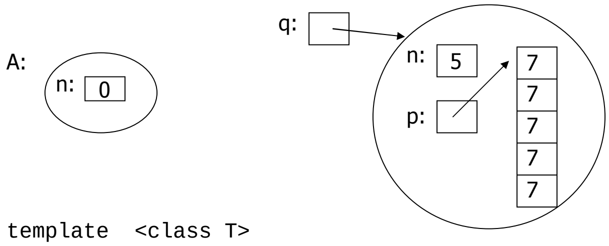
b) Implement the copy constructor and destructor functions of this class.

c) Implement the overloaded assignment operator for this class so that, regardless of the original contents of the left hand side, after assignment, it will be a copy of the right hand side.

Example Questions

A10)

a)



b)

```
template  <class T>
MT2<T>::MT2(const MT2<T> &m)
{  n=m.n;
   if (n) p=new T[n];
   for (int i=0l i<n; i++)
          *(p+i)=*(m.p+i);
}
template <class T>
MT2<T>::~MT2(void)
{  if(n) delete p[n];}
```

Example Questions

A10)

c)

```
template  <class T>
MT2<T>&MT2<T>::operator=(const MT2<T> &mt2obj)
{ if (n) delete p[n];
  n=mt2obj.n;
  if (n) p=new T[n];
  for (int i=0; i<n; i++)
         *(p+i)=*(mt2obj.p+i);
  return *this;
}
```

Example Questions

Q11) Using the following C++ definition for class Node:

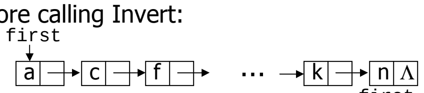
```
template <class T>
class Node
{       public:
        T data;
        Node<T> *next;
        Node(const T &item, Node<T>* ptrNext=0);
        void InsertAfter(Node<T> *p);
        Node<T> *DeleteAfter(void);
}
```

implement global function

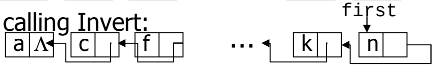
```
template <class T>
void Invert(Node<T> *first);
```

which inverts a given linked list pointed by "first", by changing only the "next" fields of the existing node. That is:

Before calling Invert:



After calling Invert:



Example Questions

A11)

```
template  <class T>
void Invert(Node<T> &*first)
{
  Node<T> *once, *bu, *sonra;
  once=NULL;
  bu=first;
  while (bu !=NULL)
         {
               sonra=bu->next;
               bu->next=once;
               once=bu;
               bu=sonra;
         }
  first=once;
}
```