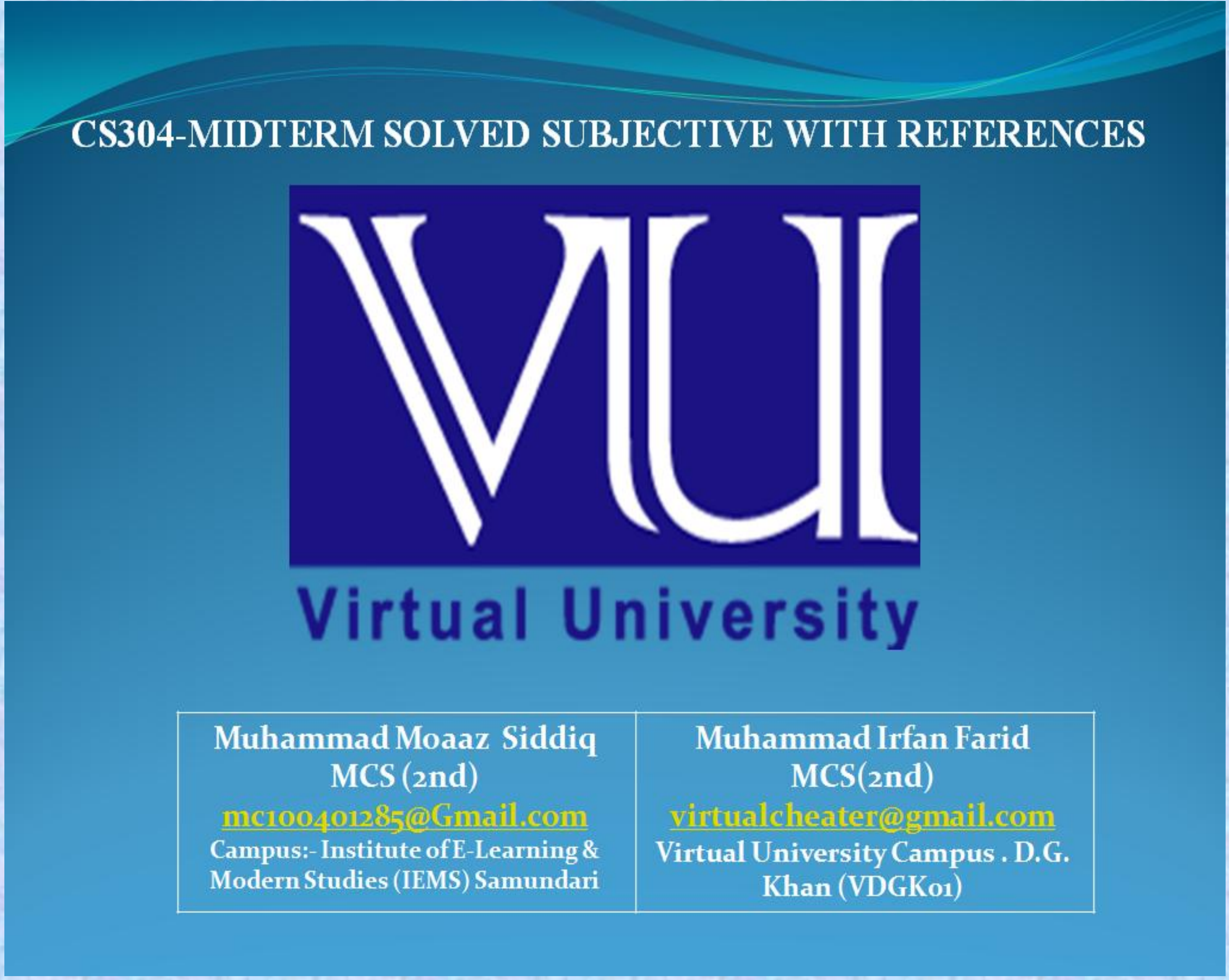
	CS304- Object Oriented Programming LATEST SOLVED SUBJECTIVES FROM MIDTERM PAPERS	NOV 29,2011
MC100401285	Moaaz.pk@gmail.com Mc100401285@gmail.com	PSMD01
MC110200885	Virtualcheater@gmail.com	VDGK01



CS304-MIDTERM SOLVED SUBJECTIVE WITH REFERENCES

Virtual University

Muhammad Moaaz Siddiq MCS (2nd) mc100401285@Gmail.com Campus:- Institute of E-Learning & Modern Studies (IEMS) Samundari	Muhammad Irfan Farid MCS(2nd) virtualcheater@gmail.com Virtual University Campus . D.G. Khan (VDGK01)

MIDTERM EXAMINATION 2011

Q.1 can constant object access the none constant member function of each class.

Answer:- (Page 106)

"Const objects can access only const member functions so chances of change of state of const objects once they are created are eliminated."

Q.2. Give at least two problems that we should check when we overloading assignments operator ("=") in string class **(unsolved)**

Q3. Give c++ code to overloaded unary "--" oprators to comples member class.

Answer:- (Page 165)

```
class Complex{
...
Complex operator -- (int);
// friend Complex operator --(const Complex &, int);
}
Complex Complex::operator -- (int){
complex t = *this;
real - = 1;
return t;
}
Complex operator -- (const
Complex & h, int){
complex t = h;
h.real - = 1;
return t;
}
```

Q4.What is simple association? Explain it with the help of example.

Answer:- (Page 49)

The two interacting objects have no intrinsic relationship with other object. It is the weakest link between objects. It is a reference by which one object can interact with some other object.

- Customer gets cash from cashier
- Employee works for a company

Q5.explain the deference between the static variable of a class with none static variable with the help of example

Answer:- click here for detail

The difference between static and non static members is only that a non static member is tied to an instance of a class although a static member is tied to the class, and not to a particular instance. That is, a static member is shared by all instances of a class although a non static member exists for each instance of class.

MIDTERM EXAMINATION 2011

write c++ code for operator() ……5 marks

Answer:- (Page 161)

Function Operator

```
class String{
...
public:
char & operator()(int);
...
};
char & String::operator()
(int pos){
assert(pos>0 && pos<=size);
return bufferPtr[pos-1];
}
int main(){
String s1(“Ping”);
char g = s1(2); // g = ‘i’
s1(2) = ‘o’;
cout << g << “\n”;
cout << str.GetString();
return 0;
}
```

Write c++ code for operator [] using string class…….5 marks…..

Answer:- (Page 160)

Subscript operator must be overloaded as member function of the class with one parameter of integer type,

```
class String{
...
public:
char & operator[](int);
...
};
char & String::operator[]( int pos){
assert(pos>0 && pos<=size);
return stringPtr[pos-1];
}
int main() {
String s1(“Ping”);
cout <<str.GetString()<< endl;
s1[2] = ‘o’;
cout << str.GetString();
return 0;
}
```

static data members.......2.mark

Answer:- (Page 108)

"A variable that is part of a class, yet is not part of any object of that class, is called static data member"

MIDTERM EXAMINATION 2011

What is interface give two real life examples?

Answer:- (Page 18)

Interface is a set of functions of an object that he wants to expose to other objects.

Interface of a Car is set of these functions
• Accelerate • Change Gear • Apply Brakes• Turn Lights On/Off

Interface of a Phone is set of these functions
• Input Number• Place Call • Disconnect Call • Add number to address book • Remove number • Update number

What is aggregation give example?

Answer:- (Page 53)

An object may contain a collection (aggregate) of other objects, the relationship between the container and the contained object is called aggregation, For Example: A room and its components like table, chair and bed etc are example of aggregation.

What is an abstract class?

Answer:- (Page 21)

Taking part of the objects and their functions that is related to the problem and ignore all other irrelative characteristic called abstraction

MIDTERM EXAMINATION 2010

Question No: 17 (Marks: 2)

Can we create an array of objects for a class having user defined constructor? Justify your answer.

Answer:- (Page 114)

There must always be a default constructor if we want to create array of objects

Question No: 18 (Marks: 2)

Friend functions increase 'Programming bugs'. What is your opinion?

Answer:-

Yes friend functions can increase programming bugs some time.

Question No: 19 (Marks: 2)

Explain two benefits of **setter functions.**

Answer:- **click here for detail**

- They help to define the variable in an interface
- To make a variable read-only
- notify other methods / classes when the value is changed

Question No: 20 (Marks: 3)

What are binary operators? Give an example of binary operators overloading using any class.

Answer:- (Page 142)

Binary Operators Overloading:

Binary operators act on two quantities.

Examples of binary operators:

Examples:

Overloading + operator:

```
class Complex{
private:
double real, img;
public:
…
Complex operator +(const Complex & rhs);
};
Complex Complex::operator +( const Complex & rhs){
Complex t;
t.real = real + rhs.real;
t.img = img + rhs.img;
return t;
}
```

The binary operator is always called with reference to the left hand argument. (Page 145)

Question No: 21 (Marks: 3)

Give c++ code to overload assignment operator for string class.

Answer:- (Page 148)

```
class String{
int size;
char * bufferPtr;
public:
String(); // default constructor
String(char *); // overloaded constructor
String(const String &); // copy constructor
…
};
String::String(){
bufferPtr = NULL;
size = 0;
}
String::String(char * ptr){
if(ptr != NULL){
size = strlen(ptr);
bufferPtr = new char[size+1];
strcpy(bufferPtr, ptr);
}
else{
bufferPtr = NULL;
size = 0;
}
}
String::String(const String & rhs){
size = rhs.size;
if(rhs.size != 0){
bufferPtr = new char[size+1];
strcpy(bufferPtr, ptr);
}
else
bufferPtr = NULL;
}
int main(){
String str1(“Hello");
String str2(“World”);
str1 = str2;78
return 0;
}
```

Question No: 22 (Marks: 5)

Writ c++ code to overload **subscript [\]** operator for **String class.**

Answer:- (Page 160)

Question No: 23 (Marks: 5)

Detect and correct compile time error(s) in the following code.

```
class  Exam
{
      char *ExamName;
      int No_of_paper;

public:
       Exam()
       {
             ExamName = "Final Term";
             No_of_paper = 5;
             }

void setname( char* name)  const
{
     ExamName = name;
     }
void setpaper(int paper)   const
{
     No_of_paper = paper;
     }
char* getname()
{
      return ExamName;
      }
int getpaper()
{
    return No_of_paper;
}
 };

int main()
{
const Exam exam1;

cout << " Exam            = "<<exam1.getname()<<endl;
cout << " Numbe of paper  = " << exam1.getpaper();

getch();
return 0;
}
```

Answer:-

Corrected Code

```
class  Exam
{
      char *ExamName;
      int No_of_paper;

public:
       Exam()
       {
             ExamName = "Final Term";
             No_of_paper = 5;
             }

void setname( char* name)
{
     ExamName=name;
     }
void setpaper(int paper)
{
     No_of_paper = paper;
     }
char* getname()
{
    return ExamName;
    }
int getpaper()
{
   return No_of_paper;
}
 };

int main()
{
Exam exam1;

cout << " Exam            = "<<exam1.getname()<<endl;
cout << " Numbe of paper  = " << exam1.getpaper();

getch();
return 0;
}
```

MIDTERM EXAMINATION 2010

Question No: 17   (Marks: 2)

If, within a class, da is a member variable, will the statement **this.da=25;** assign 25 to da? False

Answer:-

"this" is pointer, so using pointer to object we use the - > operator to access object member function and data member,

Question No: 18   (Marks: 2)

Explain the scope of 'private' member of a class with one example.

Answer:-

The scope of private data member of a class is that, only an access is allowed within the class. Only member function of class can access and modify the value of data member value. Derived class or class instance/object can't directly access. Data Member will created for each separate object in the memory and will eliminated when object will be destroy. Object could be destroy by calling its destructor. Example

```
Class arthimetic
{
   Private:
         int a,b;
   Public:
         Add()
         {
```

```
                    A+b;  // direct access by member function Correct Access
          }
};

Int main()
{
          Arthimetic c1;
          C1.a=4;// is illegal access to class private data member error will generate by compiler
}
```

Question No: 19   (Marks: 2)
Friend functions increase 'Programming bugs'. What is your opinion?
Answer:- repeat

Question No: 20   (Marks: 3)

Assuming that class X does not use any overloaded operators and has a member function to subtract two objects of X and placing the result in third object using which it has been called, write a statement that we will use to subtract two objects of class X placing the result in third object for class X. You can take any names for three objects of X.   **(unsolved)**

Question No: 21   (Marks: 3)
Give c++ code to overload unary '-- 'operator for **complex** numbers class.
Answer:- repeat

Question No: 22   (Marks: 5)
Write c++ of overloading ^ operator in **complex numbers class.**

If we have two objects of complex number class as follows,
Complex obj1,obj2;
and we write statement,
Complex obj3 = obj1 ^ obj2;
obj3 real and imaginary parts will be,

obj3.real = (obj1.real) $^{\text{obj2.real}}$ and **obj3.img = (obj1.img)** $^{\text{obj2.img}}$
Hint: You can use c++ built in function **power(x,y)** that returns the result of x ^{y}.

Answer:-

```
Class complex
{
          Private:
                    Double  val;
          Complex operator ^(complex c1)
          {
                    Return Power(val,c1.val);
          }
};
```

Question No: 22 (Marks: 5) (Another related question come in Nov-2011 mid paper)

Write c++ of overloading - operator in complex numbers class which can work in these give situation.
If we have two objects of complex number class as follows,

```
Complex obj1;
Complex obj3 = 47.4- obj2;
Complex obj4 = obj2-47.4;
```

Answer:-

Handout Page No.146

We have made them as friend so that we can write them as non member functions and they ar not called with respect to complex no. Class object instead we pass both arguments (complex no. object and double value) to this function compiler invoke them according to arguments passed. Their implementation is similar as give below.

```
Class complex
{
        private:
                double  real, img;
        public:
        friend complex operator -(double c1, complex obj);
        Complex operator -(double c1)
        {
                Complex temp;
                temp.real=real-c1;
                temp.img=img-c1;
                return temp;
        }
};
//Friend function Definition
        Complex operator -( double c1, complex obj)
        {
                Complex temp;
                temp.real=c1-real;
                temp.img=c1-img;
                return temp;
        }
```

Question No: 23 (Marks: 5)

What are Accessor Functions, Explain with an example?

Answer:- (Page93)

Accessor functions are used to access private data of the object, we provide accessor functions to get and set private data members of the class.

Example

```
class Student{
…
int rollNo;
public:
```

```
void setRollNo(int aRollNo){
rollNo = aRollNo;
}
};
Avoiding Error
void Student::setRollNo(int aRollNo){
if(aRollNo < 0){
rollNo = 0;
}
else
{
rollNo = aRollNo;
}
}
```

MIDTERM EXAMINATION
2010

Q.1

What is difference between simple association and composition?

Answer:- (PAGE 51-52)

In Association, interacting objects have no intrinsic relationship with other object. It is the weakest link between objects. While in Composition An object may be composed of other smaller objects, the relationship between the "part" objects and the "whole".

Q.2

Friend functions minimize "Encapsulation", What is your opinion?

Answer:- (PAGE 153)

Friend functions minimize encapsulation as we can access private data of any class
using friend functions,
This can result in:

- Data vulnerability
- Programming bugs
- Tough debugging

Q.3

Write three important properties of constructors?

Tell the reason why we can not overload the following four operators in c++

. , .* , :: , ?:

Considering the complex number class can we subtract two complex numbers by overloading plus "+ "Operator. Justify your answer as well.

Answer:-

Constructor Properties (Page 74)

- Constructor is a special function having same name as the class name
- Constructor does not have return type
- Constructors are commonly public members

The reason why we can not overload the following four operators in c++ **(Click here for detail)**

Because they are strictly compile time. There is a very strong argument against sizeof and unary &, since they are defined for all types, and have results which can be used in constant expressions (evaluated at "compile" time)

Q.4

How we resolve the following problems in overloading of assignment operator in string class, (explain with the help of c++ code)

If referencing

signing a string value to more than one strings in a single line like, stringobject1 = string object2 = ingobject3 = stringobject4

Answer:-

Handout Page:150

```
class String{
…
public:
        …
      String & operator = (const String &);
};
String & String :: operator = (const String & rhs){
   size = rhs.size;
   delete [] bufferPtr;
   if(rhs.size != 0){
   bufferPtr = new char[rhs.size+1];
   strcpy(bufferPtr,rhs.bufferPtr);
}
else bufferPtr = NULL;
return *this;
}
```

Q.5

Consider the class given below explain the order in which variables e,f and g will be initialized after creating object of this class,

```
class XYZ{
        int e;
        int f;
        int g;
public:
        XYZ ();
};

XYZ:: XYZ ():g(30),e(10),f(20)
{
        …
}
```

Answer:-

The value will give in the way which they are defined in the class.

E=10,f=30 and g=20

MIDTERM EXAMINATION 2010

Question No: 17 (Marks: 2) **(unsolved)**

Can we create an array of objects for a class having default constructor?. Justify your answer.

Question No: 18 (Marks: 2)

Friend functions increase ‘Data vulnerability’, what is your opinion?

Answer:- (Page 153)

Friend functions minimize encapsulation as we can access private data of any class using friend functions,
This can result in:

- Data vulnerability
- Programming bugs
- Tough debugging

Question No: 19 (Marks: 2)

Explain two benefits of setter functions.

Answer:- (Page 67)

Setters and getters functions are provided by class to access the its members it also minimizes the changes to move the objects in inconsistent state as we can write checks in our setter functions for example we can check that whether the user has entered correct age value and has not entered negative value for age.

Question No: 20 (Marks: 3)

Consider the class given below what will be the values in variables x,y and z after creating object of this class,

```
class ABC{
int x;
int y;
int z;
public:
ABC();
};
ABC::ABC():x(10),z(x),y(x)
{
…
}
```

Answer:-

X=10,z=10 and y=10

Question No: 21 (Marks: 3)

Explain what type of copy the default assignment operator "=" does when applied to objects. (shallow copy or deep copy)

Answer:- (Page 147)

if our class has any data member using dynamic memory then we had to write our own code for default constructor, copy constructor and similarly assignment operator as compiler generated version of these functions performs shallow copy that creates dangling pointer, and memory leakage issues in case of dynamic memory allocation.

Question No: 22 (Marks: 5)
What is composition? Explain it with the help of an example.
Answer:- (Page 52)
An object may be composed of other smaller objects, the relationship between the “part” objects and the “whole” object is known as Composition, and Composition is represented by a line with a filled-diamond head towards the composer object. For example a man and his body parts have the relation of “part” objects and the “whole” object is known as Composition of man.

Question No: 23 (Marks: 5)
How we can overload Stream Extraction and Insertion Operators in c++? Give example code for Complex Number Class.
Answer:- (Page 155-157)
Overloading Stream Insertion Operator

```
class Complex{
...
friend ostream & operator << (ostream & os, const Complex & c);
};
```

Stream Insertion operator

```
// we want the output as: (real, img)
ostream & operator << (ostream & os, const Complex & c){
        os << ‘(‘ << c.real
         << ‘,‘
         << c.img << ‘)’;
         return os;
}
```

Overloading Stream Extraction Operator

```
class Complex{
...
friend istream & operator >> (istream & i, Complex & c);
};
```