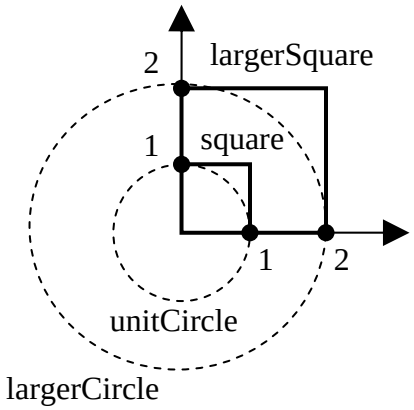
OBJECT ORIENTED PROGRAMMING

Keys to 1st Midterm Exam

(There are 2 Questions. 90-Minutes Exam, Give your answer inside the corresponding box)

Student ID	Name	Signature

Q.1) (60) You will design two classes in order to model two geometrical shapes, Circle and Rectangle, using object oriented programming approach. A typical application code is given below.
Write only required methods.




```
Circle unitCircle;
Circle largerCircle(0.0,0.0,2.0);
Rectangle square;
Rectangle largerSquare(0.0,2.0,2.0,0.0);
double f= unitCircle.area();
double g= unitCircle.circumference();
double h= square.area();
double j= square.circumference();
if (unitCircle>largerCircle) {…}
if (unitCircle==largerCircle) {…}
if (square>largerSquare) {…}
if (square==largerSquare) {…}
if (unitCircle>square) {…}
if (square==largerCircle) {…}
double left,right,top,bottom;
square.getPosition(left,top,right,bottom);
double x,y,r;
unitCircle.getPosition(x,y,r);
largerSquare.setPosition(left,top,right,bottom);
largerCircle.setPosition(x,y,r);
```

Comparisons between rectangles and circles are computed through areas.

```
class Circle {
  private:
    double x,y,r;
  public:
     Circle(double x=0.,double y=0.,double r=1.0){
        this->x= x;     this->y= y;     this->r= r;
     }
     void setPosition(double x,double y,double r){
        this->x= x;       this->y= y;       this->r= r;
     }
    void getPosition(double &x,double &y,double &r) const {
        x= this->x;   y= this->y;    r= this->r;
    }
    double area() const {return 3.1415 * r*r;}
    double circumference() const {return 2.0 * 3.1415 * r;}
} ;
bool operator>(const Circle &left, const Circle &right){
    return left.area()>right.area() ;
}
bool operator==(const Circle &left, const Circle &right){
    return left.area()==right.area() ;
}
class Rectangle {
  private:
    double left,right;
          double bottom,top;
  public:
     Rectangle(double left=0.,double top=1.,double right=1.,double bottom=0.){
        this->left= left;    this->top= top;    this->right= right;   this->bottom= bottom;
     }
     void setPosition(double left,double top,double right,double bottom){
        this->left= left;    this->top= top;    this->right= right;    this->bottom= bottom;
     }
    void getPosition(double &left,double &top,double &right,double &bottom) const {
        left= this->left;   top= this->top;   right= this->right;       bottom= this->bottom;
    }
    double area() const {return (right-left)*(top-bottom);}
    double circumference() const {return 2.0 * ((right+top-left-bottom));}
    bool operator>(const Rectangle &right) const {     return (this->area() > right.area());    }
    bool operator==(const Circle &right) const {     return (this->area() == right.area());     }
    bool operator==(const Rectangle &right) const {    return (this->area() == right.area());   }
};
bool operator>(const Circle& left, const Rectangle &right){    return (left.area() > right.area());   }
```

Download the full source code from the course web page

You may use the back of the page, but not elsewhere

Q.2) (40) What is the output of the following C++ code?

```
class Storage {
    int *hold;
  public:
    Storage (int h=0) {
      hold= new int(h);
      cout << endl << "constructor (Storage): Holding " << h ;
    }
    ~Storage () {
      cout << endl << "destructor (Storage): Releasing " << *hold ;
      delete hold;
    }
    Storage (const Storage &source) {
      hold= new int(*source.hold);
      cout << endl << "copy constructor (Storage): Holding " << *hold ;
    }
    Storage& operator= (const Storage& right) {
      cout << endl << "operator= (Storage): Releasing " << *hold ;
      delete hold;
      hold= new int(*right.hold);
      cout << endl << "operator= (Storage): Holding " << *hold ;
    }
    int& operator()() const {
      cout << endl << "operator() (Storage)" ;
      return *hold;
    }
} ;
class Array {
    Storage *array ;
    int size;
  public:
    Array(int *a,int length){
      cout << endl << "constructor (Array)" ;
      array= new Storage[length];
      for (int i=0;i<length;i++){
         array[i]()= a[i];
      }
      size= length;
    }
    int& operator[] (int i) const {
      cout << endl << "operator[] (Array)" ;
      return array[i]();
    }
    ~Array(){
      cout << endl << "destructor (Array) ";
      delete[] array;
     }
} ;
```

```
int main (int argc, char* argv[]){
①    int myArray[]={549,3615,4629};
②    int length=sizeof(myArray)/sizeof(int);
③    Array myObject(myArray,length);
④    for (int i=0;i<length;i++)
⑤        cout << endl << myObject[i] ;
⑥    return 0;
⑦ }
```

Line Number	Output
③	`constructor (Array)`
③	`constructor (Storage): Holding 0`
③	`constructor (Storage): Holding 0`
③	`constructor (Storage): Holding 0`
③	`operator() (Storage)`
③	`operator() (Storage)`
③	`operator() (Storage)`
⑤	`operator[] (Array)`
⑤	`operator() (Storage)`
⑤	`549`
⑤	`operator[] (Array)`
⑤	`operator() (Storage)`
⑤	`3615`
⑤	`operator[] (Array)`
⑤	`operator() (Storage)`
⑤	`4629`
⑥	`destructor (Array)`
⑥	`destructor (Storage): Releasing 4629`
⑥	`destructor (Storage): Releasing 3615`
⑥	`destructor (Storage): Releasing 549`

Download the full source code from the course web page.