The following code runs but can cause a serious system error. Identify and explain the error, then add code to fix it.

**See Answer in Microsoft Visual Studio Project Source.cpp**

//Objects of this class are partially filled arrays of doubles.

class PFArrayD

{

public:

PFArrayD();

//Initializes with a capacity of 50.

PFArrayD(int capacityValue);

void addElement(double element);

//Precondition: The array is not full.

//Postcondition: The element has been added.

bool full() const { return (capacity == used); }

//Returns true if the array is full, false otherwise.

int getCapacity() const { return capacity; }

int getNumberUsed() const { return used; }

void emptyArray(){ used = 0; }

//Empties the array.

double& operator[](int index);

//Read and change access to elements 0 through numberUsed - 1.

PFArrayD& operator =(const PFArrayD& rightSide);

~PFArrayD();

private:

double \*a; //For an array of doubles

int capacity; //For the size of the array

int used; //For the number of array positions currently in use

};

PFArrayD::PFArrayD() :capacity(50), used(0)

{

a = new double[capacity];

}

PFArrayD::PFArrayD(int size) : capacity(size), used(0)

{

a = new double[capacity];

}

void PFArrayD::addElement(double element)

{

if (used >= capacity)

{

cout << "Attempt to exceed capacity in PFArrayD.\n";

exit(0);

}

a[used] = element;

used++;

}

double& PFArrayD::operator[](int index)

{

if (index >= used)

{

cout << "Illegal index in PFArrayD.\n";

exit(0);

}

return a[index];

}

PFArrayD& PFArrayD::operator =(const PFArrayD& rightSide)

{

if (capacity != rightSide.capacity)

{

delete[] a;

a = new double[rightSide.capacity];

}

capacity = rightSide.capacity;

used = rightSide.used;

for (int i = 0; i < used; i++)

a[i] = rightSide.a[i];

return \*this;

}

PFArrayD::~PFArrayD()

{

delete[] a;

}

void showPFArrayD(PFArrayD parameter)

{

cout << "The first value is: "

<< parameter[0] << endl;

}

int main()

{

PFArrayD sample(2);

sample.addElement(5.5);

sample.addElement(6.6);

showPFArrayD(sample);

cout << "After call: " << sample[0] << endl;

system("pause");

return 0;

}