
Subject: Re: what is the largest Array size please? ("out of memory" error)

Posted by [nixnixnix](#) on Wed, 26 Mar 2008 12:10:45 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi Mirek,

So I've made a class to act as a container for a 2d array.

There appears to be a problem with the destructor but I can't see that there is anything wrong it. Is this something to do with the way that UPP does garbage collection?

I have an Array of Turbine objects which is defined

```
WithDeepCopy<Array<Turbine>> m_turbs;
```

and each Turbine object contains several Grid2d objects. Whenever I try to delete the Turbine objects, I get an error saying there is heap corruption.

Nick

```
class Grid2d
{
public:
    Grid2d(){m_nRows=m_nCols=0;m_ppData=NULL;}// default constructor
    Grid2d(const Grid2d& grid)    // copy constructor
    {
        m_nRows=m_nCols=0;
        m_ppData = NULL;

        if(grid.m_nRows==0 || grid.m_nCols==0)
        {
            return;
        }

        if(Dimension(grid.m_nCols,grid.m_nRows))
        {
            // copy values
            for(int i=0;i<grid.m_nCols;i++)
            {
                for(int j=0;j<m_nRows;j++)
                {
                    this->Set(i,j,grid.m_ppData[i][j]);
                }
            }
        }
    }
}
```

```

Grid2d& operator=(const Grid2d& grid)
{
    m_nRows=m_nCols=0;
    m_ppData = NULL;

    if(grid.m_nRows==0 || grid.m_nCols==0)
    {
        return *this;
    }

    if(Dimension(grid.m_nCols,grid.m_nRows))
    {
        // copy values
        for(int i=0;i<grid.m_nCols;i++)
        {
            for(int j=0;j<m_nRows;j++)
            {
                this->Set(i,j,grid.m_ppData[i][j]);
            }
        }
    }

    return *this;
}

virtual ~Grid2d()
{
    Clear();
}

void Clear()
{
    if(m_nCols==0)
        return;

    for(int i=0;i<m_nCols;i++)
    {
        delete m_ppData[i];
    }
    delete m_ppData;
    m_ppData=NULL;
    m_nRows=m_nCols=0;
}

int GetRows(){return m_nRows;}
int GetCols(){return m_nCols;}

bool SetSize(int nCols,int nRows,double val=-99999)

```

```

{return Dimension(nCols,nRows,val);}
bool Dimension(int nCols,int nRows,double val=-99999)
{
    if(m_ppData!=NULL)
    {
        if(nRows==m_nRows && nCols==m_nCols)
        {
            if(val>-99999)
            {
                for(int i=0;i<nCols;i++)
                {
                    for(int j=0;j<nRows;j++)
                    {
                        m_ppData[i][j] = val;
                    }
                }
            }
        }

        return true;
    }

    Clear();
}

m_nCols = nCols;
m_nRows = nRows;

m_ppData = new double*[nCols];

if(m_ppData==NULL)
{
    m_nRows=m_nCols=0;
    return false; // failed memory allocation so return false
}

for(int i=0;i<m_nCols;i++)
{
    m_ppData[i] = new double[nRows];

    if(m_ppData[i]==NULL)
    {
        // we failed to allocate memory so destroy everything and return false
        for( ;i<=0;i--)
        {
            delete m_ppData[i];
        }
        delete m_ppData;
        m_ppData = NULL;
    }
}

```

```

    m_nRows=m_nCols=0;
    return false;
}

if(val>-99999) // init values
{
    for(int j=0;j<m_nRows;j++)
    {
        m_ppData[i][j] = val;
    }
}

return true;
}

double* operator[](int nCol) {return GetCol(nCol);}

inline double* GetCol(int nCol)
{
    if(nCol>=0 && nCol<m_nCols)
        return m_ppData[nCol];
    else
        return m_ppData[0];
}

double Get(int nCol,int nRow) {if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)return m_ppData[nCol][nRow];}
void Set(int nCol,int nRow,double val) {if(nRow>=0 && nCol>=0 && nRow<m_nRows &&
nCol<m_nCols)m_ppData[nCol][nRow]=val;}

virtual void Serialize(Stream& s)
{
    int nRows = m_nRows;
    int nCols = m_nCols;

    s % nCols % nRows;

    if(s.IsLoading())
    {
        if(!Dimension(nCols,nRows))
            return;
    }

    for(int i=0;i<m_nCols;i++)
    {
        for(int j=0;j<m_nRows;j++)
        {

```

```
        s % m_ppData[i][j];  
    }  
}  
}
```

protected:

```
// current size of grid  
int m_nRows;  
int m_nCols;  
  
// data values  
double** m_ppData;  
};
```
