New Added Features: Sorting & Searching

The following code snippet demonstrates the current new added feature functionalities in mixutil.h

```cpp
#if !defined(REQUIRE_H)
        #include "D:\MIXUTIL\User\Libs\mixutil.h"
#endif


struct _xItem
{
        _xItem() = default;
        ~_xItem() = default;

        _xItem(int const _brId, std::string const& _brName)
        : _brandId(_brId), _brandName(_brName)
        {

        }

        operator char() const {
            return this->_brandName[0];
        }

        operator int() const {
            return this->_brandId;
        }

         int _brandId;
         std::string _brandName;
};


void _printItems(const std::vector<_xItem>& _vitemX)
{
   for (auto const& _ix : _vitemX)
   {
        RPRINT(_ix._brandId); RPRINT("=>"); RPRINT(_ix._brandName);
        RET;
   }
}


int main()
{

   std::vector<_xItem> vtx = {
     {201,"OLED Touchscreen Panel"},
     {110,"Universal Wireless Charger"},
```

```cpp
   {105,"Bluetooth Speaker"},
   {98, "Office Desk Stand"},
   {115,"LED Light Tube"}
};

    _xItem xi;
    std::vector<_xItem>::iterator iGetX;

    // sort the data elements based on '_brandId' (int)
    mix::generic::
    fast_sort(vtx.begin(), vtx.end(), mix::generic::numLess());

    _printItems(vtx);

    RET;

    xi._brandId = 110;

    // search item which Id = 110
    if (mix::generic::
       vector_search(vtx.begin(), vtx.end(), xi, mix::generic::numLess(),
                 iGetX))
    {
       RPRINT(iGetX->_brandId); RPRINT("=>");
       RPRINT(iGetX->_brandName);
       RET;
    }
    else PRINT("Item not Found !!");

    RET;

    // sort the data elements based on '_brandName' (char)
    mix::generic::
     fast_sort(vtx.begin(), vtx.end(),
         mix::generic::NLess<char>());

    _printItems(vtx);

    RET;

    xi._brandName = "OLED Touchscreen Panel";
    char _c = xi._brandName[0];

    // search item which brand name = 'Office Desk Stand'
    if (mix::generic::
       vector_search(vtx.begin(), vtx.end(), _c,
                 mix::generic::NLess<char>(), iGetX))
    {
      RPRINT((iGetX - 1)->_brandId); RPRINT("=>");
           RPRINT((iGetX - 1)->_brandName); RET;

      RPRINT(iGetX->_brandId); RPRINT("=>");
           RPRINT(iGetX->_brandName); RET;

      RPRINT((iGetX + 1)->_brandId); RPRINT("=>");
```

```cpp
                RPRINT((iGetX + 1)->_brandName); RET;
        RET;
    }
    else PRINT("Item not Found !!");



    vtx.clear();

    RET2();
    return -1;
}
```