Generic Programming

C++ has Templates

- C has void* and function pointers
- How do we write a Linked List that accepts **any** data type?
- How do we write a Linked List that accepts **any** data type and keeps the list in **sorted** order?
- How do we apply the same function to every element in a list? Print?

void*

```
typedef struct Node* NodePtr;

typedef struct Node
{
  void* data;
  unsigned int size;
  NodePtr *psNext;
} Node;

NodePtr pNode;
                          // what value does sizeof return?
pNode = (NodePtr) malloc(sizeof(Node));
```

The list does not know what *type* **data** points to.

Let's write lstInsert()

Compare

size_t
unsigned *??*
in stdlib.h via
stddef.h

- If we insert two ints, how do we compare them?

- How do we compare two void* items?
  - no data type information

```
#include <string.h>
int memcmp(void* ptr, void*  ptr2, size_t size);
int memcpy(void* dest, void* src, size_t size);
```

size_t

- Look in a C Eclipse Project | Includes | /usr/lib64/gcc/x86_64-suse-linux/4.5/include | stddef.h

- line 208

```
#define __SIZE_TYPE__ long unsigned int

typedef __SIZE_TYPE__ size_t;
```

Function Pointers

```
returnType (*name)(paramType ...)

int (*foo)(int);

int negate(int x)
{
  return -x;
}

foo = &negate;
(*foo)(3);
```

Can we print every element of the list?

We know the data type stored in the void*

The List does not know the data type!

```
// assume we have ints in the list
void print(void* pData)
{
   printf("%d ", * ((int*) pData) );
}
```

Sorting

- How can we tell which int is larger?

```
ERR_CODE lstInsertSorted(ListPtr psList,
                         void* pData,
                         unsigned int size,
int (*compare)(const void*, const void*));
```

Sorting arrays

```
#include <stdlib.h>

void qsort(void *base, size_t nmemb,
    size_t size,
    int(*compar)(const void *, const void *));
```