**March 3, 2021**

**1. Template (continued)**

**\* Main purpose is to utilize generic data types for reducing programming time**

* **Several applications: template function, template class, template struct**
* **It does not reduce memory usage**
* **Template function**

**template <class T>**

**T foo(T x, int y)**

**{ … }**

**Int main( )**

**{**

**Char a = ‘A’, c;**

**Int b = 10;**

**c = foo(a, b);**

**…**

**}**

**template <class T>**

**class D**

**{**

**T x;**

**Public:**

**D( ) { … }**

**…**

**};**

**Int main( )**

**{**

**D <int> a;**

**D <double> b;**

**…**

**}**

**template <class T>**

**struct E**

**{**

**T IDnumber;**

**double x;**

**};**

**Template <class T>**

## Array<T>::Array( unsigned sz )

## { values = new T [sz];

## size = sz;

## }

**template<class T>**

T& Array<T>::operator[](unsigned i)

{

return values[I];

}

**Int main( )**

**{**

**E <int> a;**

**E <Student> b;**

**…**

**}**

**2. Exceptions**

**\* Exception Handler: a special function that can be used to do post-processing when some undesired events happen.**

* **Syntax**

**+ try block**

**+ catch block**

**+ throw action**

**Exception handling in C++ is not very good, compared to Java and C#.**

**…**

**try**

**{**

**… // your segment of code that may cause**

**// problems**

**}**

**catch( int x )**

**{**

**// Exception handler**

**}**

**Catch( char x)**

**{ …… }**

**Catch( …) // to catch everything**

**{**

**}**

**3. Pointer**

* **Concept:**

**Int x; double y; char z[100]; Student w;**

**Sizeof(int) sizeof(double) sizeof(Student)**

**All the beginning memory addresses can be represented by an integer.**