Today 2-3pm - office hours cancelled

Friday - 10am Connor

1pm Duncan

I will be at the OCCE - office hours cancelled.

CS 162
Intro to CS II

Exceptions and Templates

OSU Oregon State University

Error Handling

C/C++

- Prevent it from happening
  - Checks before using it

checked the denom before a division

- Let it happen and catch it

C++

  - Exceptions

Exception Handling

- try { … } ~~block~~ code
- catch(~~exception~~ general/base class type &e) { … } ~~block~~ code your choice
- Existing Exceptions:
  - Out of Range… string
  - Bad memory allocation… new
- Order matters!!!

if you have a try you have to have a catch,

Throwing Your Own Exception

try { …

throw variable/object;

… }

catch(var_obj_type e) { … }

Keyword

type of

ignore code after throw inside the try

resume here

Programming Demo

access.engr.orst.edu - PuTTY

```
1  #include <stdexcept>
2  //#include <exception> //Don't need this when you include stdexcept
3  #include <iostream>
4  #include <string>
5  using namespace std;
6
7  int main(){
8     string s;
9     int n;
10
11    try {
12       throw n;
13       cout << s.at(0) << endl;
14       cout << s[0] << endl;
15    }
16    catch(int &e) { cout << s[0] << "int thrown!" << endl; }
17    catch(exception &e) {
18       cout << "General issue!" << endl;
19    }
20    catch(out_of_range &e) {
21       cout << "You accessed something out of range!" << endl;
22    }
23    cout << "here";
                                                    1,4          Top
```

Function Throw Exceptions

```
return_type func_name(…) throw (type1, type2, …);
int main() {
   try { func_name(); }
   catch(type1 e) { … }
}
return_type func_name(…) {
   …
   throw type1;
   …
}
```

Why Function Templates?

```
//at least C++ has overload
void swap(int &, int &);
void swap(char &, char &);
…
void swap(int &a, int &b){
  int temp = a;
  a=b;
  b=temp;
}
void swap(char &a, char &b) {
  char temp = a;
  a=b;
  b=temp;
}
```

3

Function Template…

```
//Have to have this header
template<class T>
void swap(T &, T &);
…
template<class T>
void swap(T &a, T &b){
  T temp = a;
  a=b;
  b=temp;
}
```

typename

When can you get into trouble?

```
//Have to have this header
template<class T>
void func(T, T, int);
…
template<class T>
void func(T a[], T b[], int size){
  //a is already a reference
  //what if we wanted to swap values in arrays
}
```

Why make a class templates?

- What example can we use? Vectors!!!!

Using a vector

access.engr.orst.edu - PuTTY

```
 1 #include <stdexcept>
 2 //#include <exception> //Don't need this when you include stdexcept
 3 #include <iostream>
 4 #include <string>
 5 #include <vector>
 6 using namespace std;
 7
 8 int main(){
 9    string s;
10    int n;
11    vector <int> v(2); //make a vector to hold 2 ints
12
13    v[0]=1;
14    v[1]=2;
15    cout << v.size() << endl;
16    v.push_back(3);  //push the number 3 to the back of list
17    cout << v.size() << endl;
18
19    try {
20       throw n;
21       cout << s.at(0) << endl;
22       cout << s[0] << endl;
23    }
-- INSERT --                                          16,60          Top
```

Class Templates

```
//Have to have this header
template<class T>
class vector {
  public:
    vector();
    ~vector();
    void push_back(T);
  private:
    T *v;
};
```

typename

Class Templates

```
//Have to have this header
vector::vector(){
  v=NULL;
}
vector::~vector(){
  delete [] v;
}
template<class T>
void push_back(T element){
  …
}
```

Oregon State University