ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

ΑΝΟΙΚΤΑ ακαδημαϊκά μαθήματα ΠΠ

Οντοκεντρικός Προγραμματισμός

Ενότητα 7: C++ TEMPLATES, ΥΠΕΡΦΟΡΤΩΣΗ ΤΕΛΕΣΤΩΝ, ΕΞΑΙΡΕΣΕΙΣ

Χειρισμός Εξαιρέσεων

ΔΙΔΑΣΚΟΝΤΕΣ: Ιωάννης Χατζηλυγερούδης, Χρήστος Μακρής

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Χειρισμός Εξαιρέσεων

Εξαιρέσεις

- Εξαίρεση
  - Δείχνουν ότι κάποιο πρόβλημα προέκυψε στην εκτέλεση του προγράμματος
  - Κάτι μη φυσιολογικό
- Χειρισμός Εξαιρέσεων
  - Resolve exceptions
  - Το πρόγραμμα μπορεί να συνεχίσει την λειτουργία
    - Controlled termination
  - Για δημιουργία fault-tolerant programs

Try – Catch Blocks

- Κώδικας C++

```
try {
  κώδικα που μπορεί να προκαλέσει εξαίρεση
}
catch (exceptionType){
  κώδικας για χειρισμό εξαιρέσεων
}
```

  - Το **`try`** μπλοκ περιέχει κώδικα ο οποίος μπορεί να προκαλέσει εξαίρεση
  - Τα **`catch`** μπλοκ (1 ή περισσότεραα):
    - Λαμβάνουν και χειρίζονται τις εξαιρέσεις
    - Μέσω παραμέτρου μπορούν να προσπελάσουν το αντικείμενο εξαίρεσης.

Παράδειγμα

```
#include <iostream>
#include <stdexcept>
using namespace std;
class DivideByZeroException : public out_of_range {
  public:
    DivideByZeroException(int n, string m)
              : mumerator(n), out_of_range(m){}
    int mumerator;
};

double divide( int numerator, int denominator ) {
  if ( denominator == 0 )
    throw DivideByZeroException(numerator,"attempted to divide with zero");
  return static_cast< double >( numerator ) / denominator;
}

int main(){
  int a,b;
  try{
    while ( cin >> a >> b )
      divide(a,b);
  }
  catch (out_of_range e){
    cout << e.what();
  }
}
```

Αν και δεν είναι υποχρεωτικό, εδώ φτιάχνουμε δικό μας τύπο εξαίρεσης, κληρονομώντας από τον υπάρχων τύπο out_of_range.
Μέσω του δημιουργού μπορούμε να ορίσουμε την πληροφορία που θα αποθηκεύουμε στην εξαίρεση μέσω των ορισμάτων του. (πχ εδώ μπορούμε να κρατάμε τον αριθμητή της πράξης).
Στον δημιουργό της κλάσης out_of_range μπορούμε να περνάμε μήνυμα σχετικό με το σφάλμα το οποία θα είναι προσπελάσιμο μέσω της μεθόδου what() που έχει η κλάση exception και παράγωγες της.

Με κλήση του δημιουργού, δημιουργούμε εξαίρεση του τύπου που φτιάξαμε και την πετάμε με την εντολή throw.

Στο catch κομμάτι χειριζόμαστε εξαιρέσεις τύπου out_of_range ή παράγωγες της (όπως αυτή που φτιάξαμε) .

Ροή εκτέλεσης

- Σημείο έγερσης εξαίρεσης (Throw point)
  - Το σημείο στο μπλοκ **`try`** όπου εγείρεται εξαίρεση
  - Αν χειριστεί η εξαίρεση
    - Το πρόγραμμα προσπερνάει τον υπόλοιπο κώδικα του **`try`** μπλοκ
    - Συνεχίζεται η λειτουργία μετά τα **`catch`** μπλοκ
  - αν δεν χειριστεί η εξαίρεση
    - Τερματίζεται η λειτουργία
- Αν δεν προκύψει εξαίρεση
  - Το πρόγραμμα αγνοεί τα **`catch`** blocks.

Τεχνικές Χειρισμού Εξαιρέσεων

- Αγνόηση εξαιρέσεων
  - Τυπικό για προσωπικό (όχι εμπορικό) λογισμικό
  - Το πρόγραμμα μπορεί να αποτυγχάνει
- Τερματισμός Προγράμματος
  - Συνήθως είναι κατάλληλο
  - Δεν είναι κατάλληλο για κρίσιμες εφαρμογές
- Set error indicators
  - Unfortunately, may not test for these when necessary
- Test for error condition
  - Call exit (**<cstdlib>**) and pass error code

Άμεση έγερση εξαιρέσεων

- Εντολή **`throw`**
  - Εγείρει μια εξαίρεση
    - Χρήση όταν προκύπτει το σφάλμα
  - Μπορούμε με «πετάξουμε» με την throw σχεδόν οτιδήποτε (αντικείμενα-εξαιρέσεις, built-in τύπους όπως integer, κτλ.)
    - **`throw myObject;`**
    - **`throw 5;`**
- Αντικείμενα-Εξαιρέσεις
  - Έχουν ως κλάση βάσης την **`exception`** (**`<exception>`**)
  - Ο δημιουργός μπορεί να έχει ένα αλφαριθμητικό για περιγραφή του σφάλματος.
  - Η περιγραφή μπορεί να ανακτηθεί μέσω της μεθόδου **`what()`**

Επανέγερση

- Επανέγερση εξαίρεσης (rethrowing)
  - Χρησιμοποιείται σε catch μπλοκ, όταν δεν μπορεί να χειριστεί η εξαίρεση ώστε να επανεγερθεί
  - Can rethrow exception to another handler
    - Goes to next enclosing **`try`** block
    - Corresponding **`catch`** blocks try to handle
- Για να ξαναπετάξουμε εξαίρεση μέσα σε catch block καλούμε την εντολή **`throw`** (χωρίς όρισμα)

Λίστα εξαιρέσεων συνάρτησης

- Ορίζουμε την λίστα των εξαιρέσεων που μπορεί να εγερθούν στην συνάρτηση αυτήν
  - Also called throw list

```
int someFunction( double value )
  throw ( ExceptionA, ExceptionB, ExceptionC )
{
   // σώμα συνάρτησης
}
```

  - Η συνάρτηση μπορεί να πετάξει εξαιρέσεις τύπου **`ExceptionA`**, **`ExceptionB`**, and **`ExceptionC`** (ή παράγωγες αυτών)
    - Αν προκύψει άλλου είδους εξαίρεση (και δεν χειριστεί μέσα στο σώμα της συνάρτησης με κάποιο catch block) προκύπτει απροσδόκητο σφάλμα και τερματίζει το πρόγραμμα
  - Αν δεν ορίσουμε λίστα throw, μπορεί να πετάξει οποιαδήποτε
  - Αν ορίσουμε κενή λίστα throw, δεν μπορεί να πετάξει καμία εξαίρεση

Παραδείγματα

```
void function() throw(int){
  throw 5;
}

int main(){
  try{ function(); }
  catch(int){
     cout << " handled";
  }
}
```

handled
(εγείρεται εξαίρεση αποδεκτού τύπου στην function).
Η main έχει κατάλληλο catch μπλοκ χειρισμού του τύπου οπότε χειρίζεται και τερματίζει κανονικά η λειτουργία

```
void function() throw(int){
  throw 'k';
}

int main(){
  try{ function(); }
  catch(int){
     cout << " handled";
  }
}
```

Τερματισμός με σφάλμα
Μη αποδεκτός τύπος εξαίρεσης.
Δεν έχει οριστεί ο τύπος char στην λίστα throw της συνάρτησης.

```
void function() throw(int){
  try{
    throw 'k';
  }
  catch(...){
    cout << "handled internally"; }
}
int main(){
  try{ function(); }
  catch(int){
     cout << " handled";
  }
}
```

handled internally
(Εγείρεται εξαίρεση μη αποδεκτού τύπου, ωστόσο χειρίζεται εσωτερικά).
Η main δεν «ενημερώνεται» ότι προέκυψε εξαίρεση στην function

Παραδείγματα

```
void function() throw(int, char){
  try{
    throw 'k';
  }
  catch(...){
    cout << "handled internally";
    throw;
  }
}

int main(){
  try{ function(); }
  catch(int){
     cout << " handled";
  }
}
```

handled internally
Τερματίζει με σφάλμα
(Η εξαίρεση συλλαμβάνεται αρχικά εσωτερικά, ωστόσο με την throw την πετάει ξανά στο παραπάνω επίπεδο (main) , το οποίο δεν έχει κατάλληλο catch χειρισμού της.

```
void function() throw(int, char){
  try{
    throw 'k';
  }
  catch(...){
    cout << "handled internally";
    throw;
  }
}

int main(){
  try{ function(); }
  catch(char ch){
     cout <<endl<< ch << " handled";
  }
}
```

handled internally
k handled
Η εξαίρεση χειρίζεται και εσωτερικά και στην main

Πρόσθετο Υλικό

- Μελετήστε και τα παραδείγματα από το **Κεφάλαιο 17** του βιβλίου:
  «C++ How to Program, 9/e Paul & Harvey Deitel»
  http://media.pearsoncmg.com/ph/esm/deitel/cpp_htp_9/code_examples/Code_Examples.zip

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στο πλαίσιο του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Πανεπιστήμιο Αθηνών**» έχει χρηματοδοτήσει μόνο την αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Σημείωμα Ιστορικού Εκδόσεων Έργου

Το παρόν έργο αποτελεί την έκδοση 1.0.1

Σημείωμα Αναφοράς

Σημείωμα Αδειοδότησης

Διατήρηση Σημειωμάτων

Οποιαδήποτε αναπαραγωγή ή διασκευή του υλικού θα πρέπει να συμπεριλαμβάνει:

- το Σημείωμα Αναφοράς
- το Σημείωμα Αδειοδότησης
- τη δήλωση Διατήρησης Σημειωμάτων
- το Σημείωμα Χρήσης Έργων Τρίτων (εφόσον υπάρχει)

μαζί με τους συνοδευόμενους υπερσυνδέσμους.

Σημείωμα Χρήσης Έργων Τρίτων

- Οι διαφάνειες βασίζονται στο βιβλίο «C++ How to Program, 8th Edition, Harvey M. Deitel, Paul J. Deitel, Prentice Hall.»