Προγραμματισμός Υπολογιστών με C++

ΕΞΑΙΡΕΣΕΙΣ

Γεώργιος Παπαϊωάννου (2013-17)
gepap@aueb.gr

Περιεχόμενο Παρουσίασης

- Περιγραφή:
  - Οι εξαιρέσεις στη C++
  - Δήλωση δικών μας εξαιρέσεων
  - Αναχαίτιση εξαιρέσεων
  - Δημιουργία εξαιρέσεων

- Τελευταία ενημέρωση: Δεκέμβριος 2017

Οι Εξαιρέσεις στη C++

- Και η C++ υποστηρίζει εξαιρέσεις
- Δεν είναι απαραίτητο να κληρονομούν κάποια συγκεκριμένη κλάση, μπορούν να είναι οποιουδήποτε τύπου, ακόμα και κάποιου βασικού (π.χ. int)
- Τις αναχαιτίζουμε κατά τα γνωστά μέσα σε try {} catch () {} blocks

Παράδειγμα

```
char safeReadByte(istream & source) {
   try {
     if (!source) throw int(-1);     // πετάμε εξαίρεση
   }
   catch (int e) {                   // εδώ, την πιάνουμε τοπικά
     cerr << "error accessing " << typeid(source).name() << endl;
     throw int (-1);                 // και την ξαναπετάμε
   }                                 // Η ροή ελέγχου πηγαίνει αμέσως
   return source.get();              // στο catch του πιο έξω try
}                                    // Δεν εκτελείται η return
int main() {
   try  {
      ifstream src = ifstream("mapa.txt");
      safeReadByte(src);
   }
   catch (int ex) {
      cout << "Unable to read" << endl;
   }
}
```

Διαχείριση Πολλαπλών Εξαιρέσεων

```
try {
  // code here
}
catch (int param)
{
   cout << "int exception";
}
catch (char param)
{
   cout << "char exception";
}
catch (...)
{
   cout << "default exception";
}
```

Σημαίνει: Πιάσε όλες τις υπόλοιπες εξαιρέσεις που δεν αναφέρονται παραπάνω

Η exception

- Αν επιθυμούμε, μπορούμε να χρησιμοποιούμε ως εξαίρεση αυτή που ορίζει η C++ (κλάση exception)
- Μπορούμε επίσης να την επεκτείνουμε:

```
#include <exception>

class FileException : public exception {
public:
   virtual const char* what() const noexcept(true) {
    return "error accessing stream";
   }
}

char safeReadByte(istream &source) {
   if (!source)
      throw FileException();
   return source.get();
}
```

Παράγωγες εξαιρέσεις (std)

exception	description
bad_alloc	thrown by new on allocation failure
bad_cast	thrown by dynamic_cast when it fails in a dynamic cast
bad_exception	thrown by certain dynamic exception specifiers
bad_typeid	thrown by typeid
bad_function_call	thrown by empty function objects
bad_weak_ptr	thrown by shared_ptr when passed a bad weak_ptr

exception	description
logic_error	error related to the internal logic of the program
runtime_error	error detected during runtime

Προσδιορισμός των Εξαιρέσεων μιας Συνάρτησης

- Μπορούμε αν θέλουμε, στη δήλωση μιας συνάρτησης η μεθόδου να απαγορεύσουμε να πετάει εξαιρέσεις

```
char unsafeReadByte(istream &source) noexcept(true)   // δεν πρέπει να
                                                      // πετάει εξαίρεση
```

- Είναι σφάλμα να χρησιμοποιούμε το noexcept(true) αν:
  - Εσωτερικά η συνάρτηση/μέθοδος καλεί άλλη που πετάει εξαίρεση

Πότε Χρησιμοποιούμε Εξαιρέσεις (1)

- Καταρχήν, σε μεγάλο βαθμό **οι εξαιρέσεις καταλύουν την αναγνωσιμότητα του κώδικα**:
  - Όταν ριχθεί μια εξαίρεση, η ροή ελέγχου φεύγει από τον τοπικό έλεγχο και πηγαίνει στην επόμενη catch() που την αναχαιτίζει
  - Αυτή η catch μπορεί να είναι πολλά επίπεδα τη stack κλήσης συναρτήσεων πιο πάνω
  - Χάνεται η δομημένη μορφή του κώδικα

Πότε Χρησιμοποιούμε Εξαιρέσεις (2)

- **Οι εξαιρέσεις είναι ακριβές όταν συμβούν** (~ 20 φορές το κόστος ενός απλού ελέγχου διακλάδωσης) χωρίς να υπολογίσουμε το κόστος της οπισθοδρόμησης της stack (stack unrolling)
- Οι εξαιρέσεις είναι χρήσιμες μόνο όταν η τρέχουσα συνάρτηση:
  - **Δε μπορεί να χειριστεί το συμβάν** (π.χ. δεν έχει αρκετά δεδομένα)
  - Η τρέχουσα συνάρτηση **δεν έχει την αρμοδιότητα να απαντήσει**