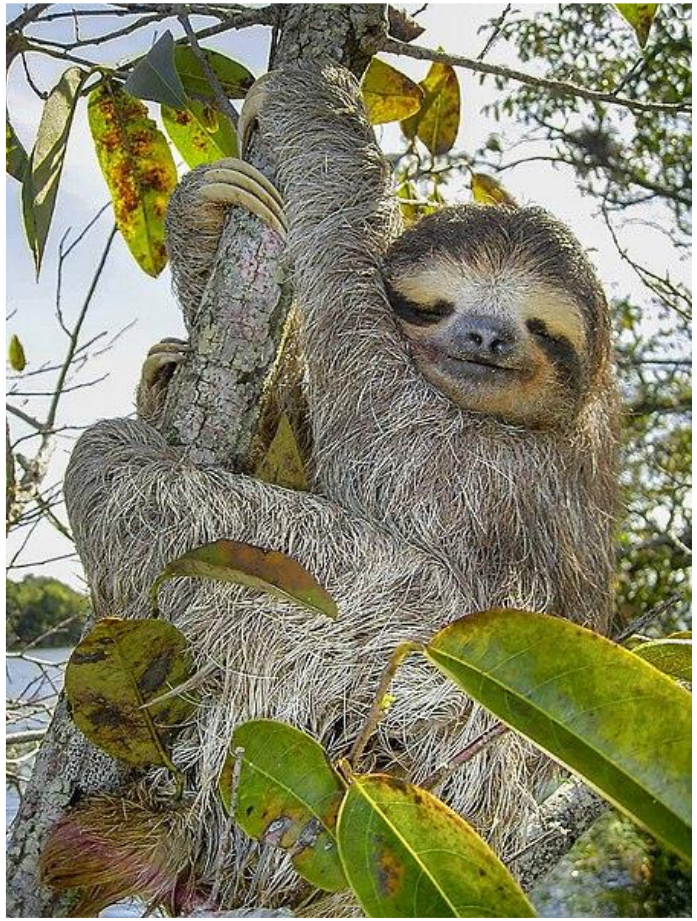
What should code do?

Basics

- Not crash
- Not cause damage
- Not lose data
- Not …



Requirements

"For any X, $|X - \boldsymbol{sqrt}(X)^2| < \varepsilon$"

Requirements

“The ‘pay’ button’s background color must be #FF0000”

Requirements

"When the 'insert in database' method returns, the data must persist, *even if power goes out or one disk is corrupted*"

Requirements

“Customers should enjoy using the software”

Requirements

- Details, formality optional
- Any level: function, module, UI…
- Usually not explicit
- Always exist

Specification

- Evolution of requirements
- Formal
- Unambiguous
- Can be verified

	Requirement	Specification
Detailed?	Maybe	Yes
Formal?	Maybe	Yes
Objective?	Maybe	Yes
Author	User	Developer

Formal verification

- Proof that code is correct
- Hard
- Time-consuming

Formal verification

```
if (bb == 0 || kh != key_hash) {
  //@ if (bb != 0) no_hash_no_key(ks, khs, k, index, hsh);
  //@ else no_bb_no_key(ks, bbs, index);
  if (chn == 0) {
    //@ assert length(chnlist) == capacity;
    //@ buckets_keys_chns_same_len(buckets);
    //@ assert length(buckets) == capacity;
    //@ no_crossing_chains_here(buckets, index);
    //@ assert nil == get_crossing_chains_fp(buckets, index);
    //@ key_is_contained_in_the_bucket(buckets, capacity, hsh, k);
    //@ assert up_to(nat_of_int(i), (byLoopNthProp)(ks, (neq)(some(k)), c
    //@ assert up_to(nat_of_int(i), (byLoopNthProp)(ks, (neq)(some(k)), c
    //@ assert up_to(succ(nat_of_int(i)), (byLoopNthProp)(ks, (neq)(some
    //@ assert up_to(nat_of_int(i+1), (byLoopNthProp)(ks, (neq)(some(k)
    //@ assert buckets != nil;
    //@ chains_depleted_no_hope(buckets, i, loop_fp(hsh(k), capacity), k
    //@ assert !hmap_exists_key_fp(hm, k);
    //@ close hmapping<kt>(kpr, hsh, capacity, busybits, kps, k_hashes, h
    //@ close buckets_ks_insync(chns, capacity, buckets, hsh, ks);
    return -1;
  }
  //@ assert length(ks) == capacity;
}
```

```
lemma void no_hash_no_key<kt>(list<option<kt>> ks, list<unsigned> hashes,
                              kt k, int i, fixpoint (kt,unsigned) hash)
  requires hash_list(ks, hashes, hash) &*&
           nth(i, hashes) != hash(k) &*& 0 <= i &*& i < length(ks);
  ensures nth(i, ks) != some(k);
  {
    switch(ks) {
      case nil:
        assert hashes == nil;
        return;
      case cons(kh,kt):
        assert hashes != nil;
        if (i == 0) {
          assert nth(i, ks) == kh;
          if (kh == some(k)) {
            assert head(hashes) == hash(k);
            nth_0_head(hashes);
            assert nth(i, hashes) == head(hashes);
            assert nth(i, hashes) == hash(k);
          }
        } else {
          nth_cons(i, tail(hashes), head(hashes));
          cons_head_tail(hashes);
          assert nth(i, hashes) == nth(i-1,tail(hashes));
          no_hash_no_key(kt, tail(hashes), k, i-1, hash);
        }
    }
  }
```

Testing

- Check behavior in known cases
- Give confidence in code
- Not a proof!

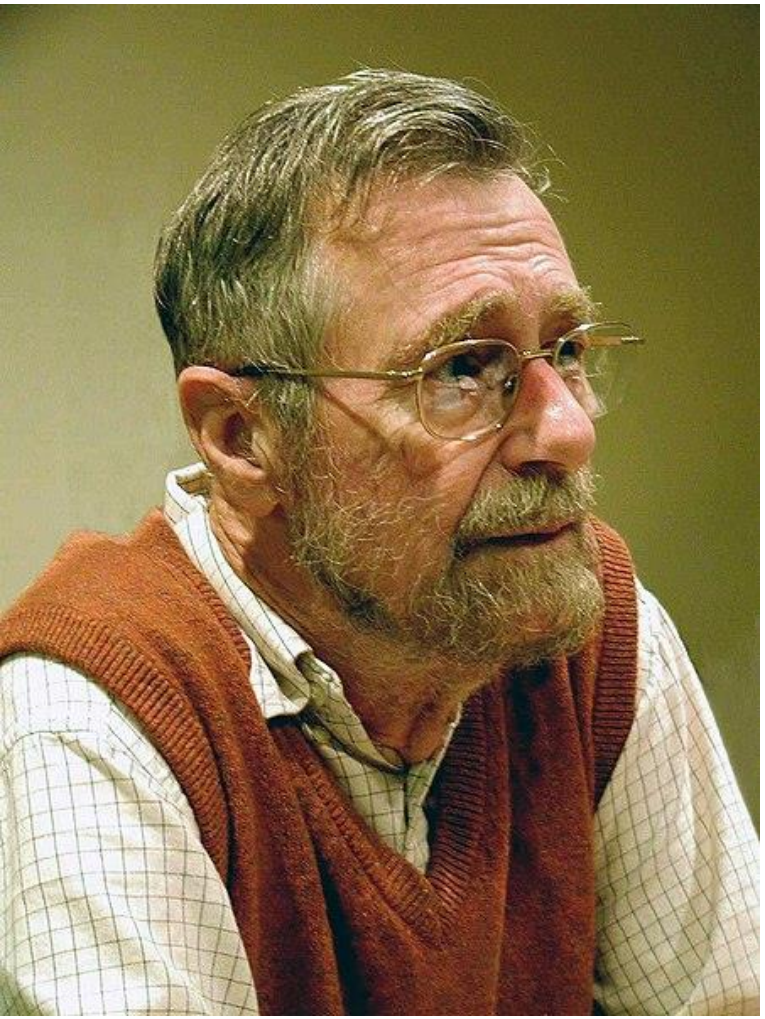


Image source: Wikipedia (CC BY-SA 3.0)

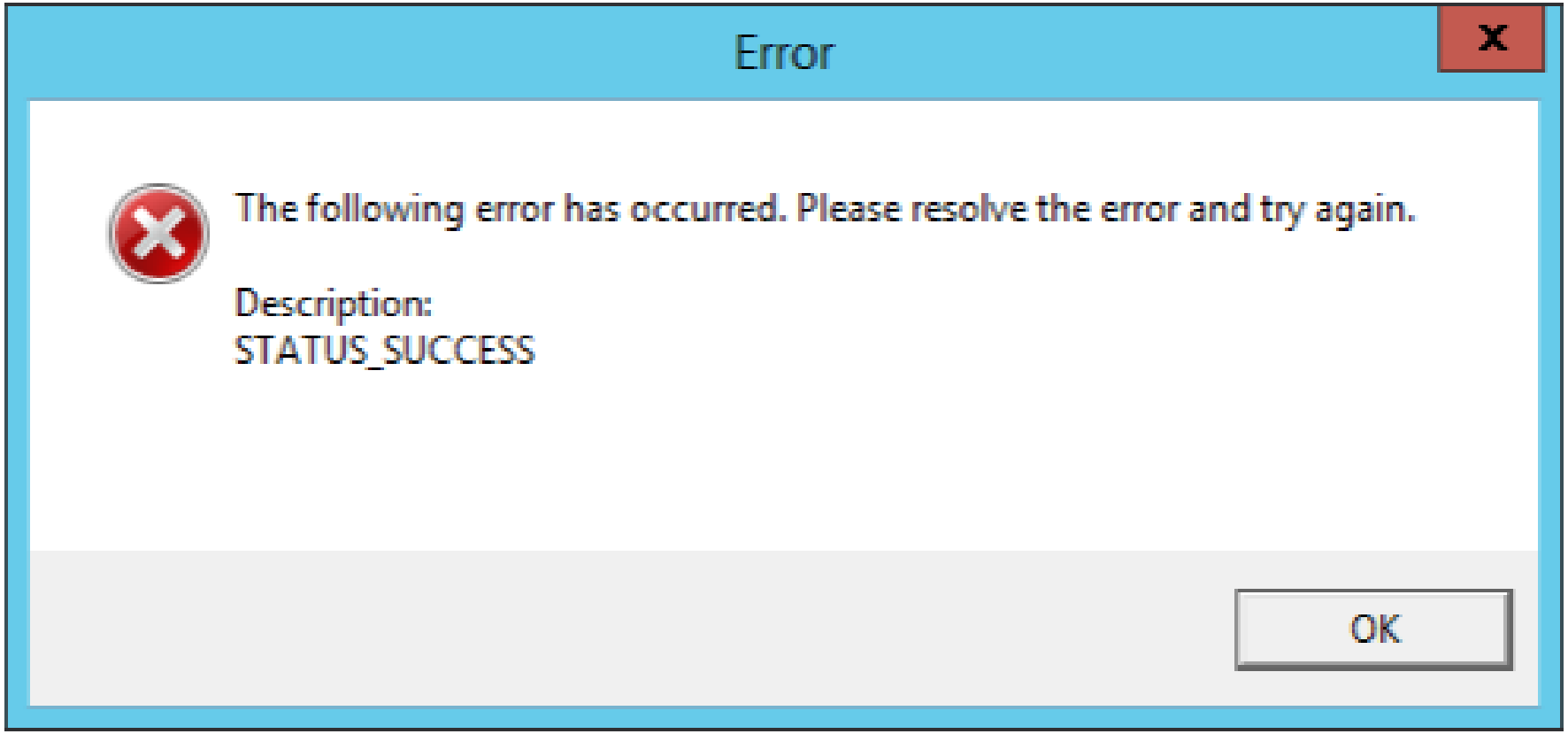
"Program testing can be used
to show the presence of bugs,
but never to show their absence!"

Edsger W. Dijkstra

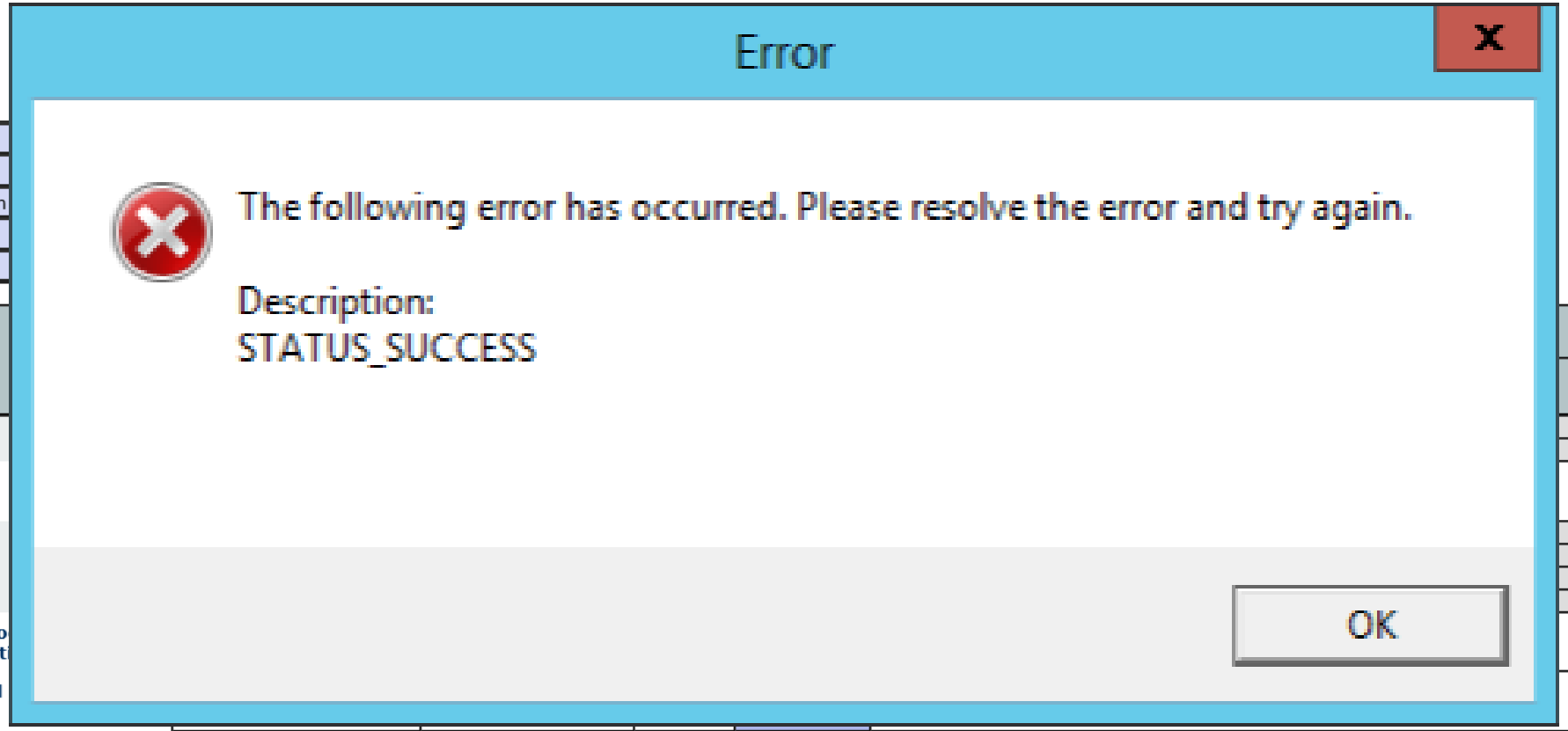
Testing

- Why?
- How?
- What?
- When?
- Evaluating tests

Why?


Error
The following error has occurred. Please resolve the error and try again.
Description:
STATUS_SUCCESS
OK

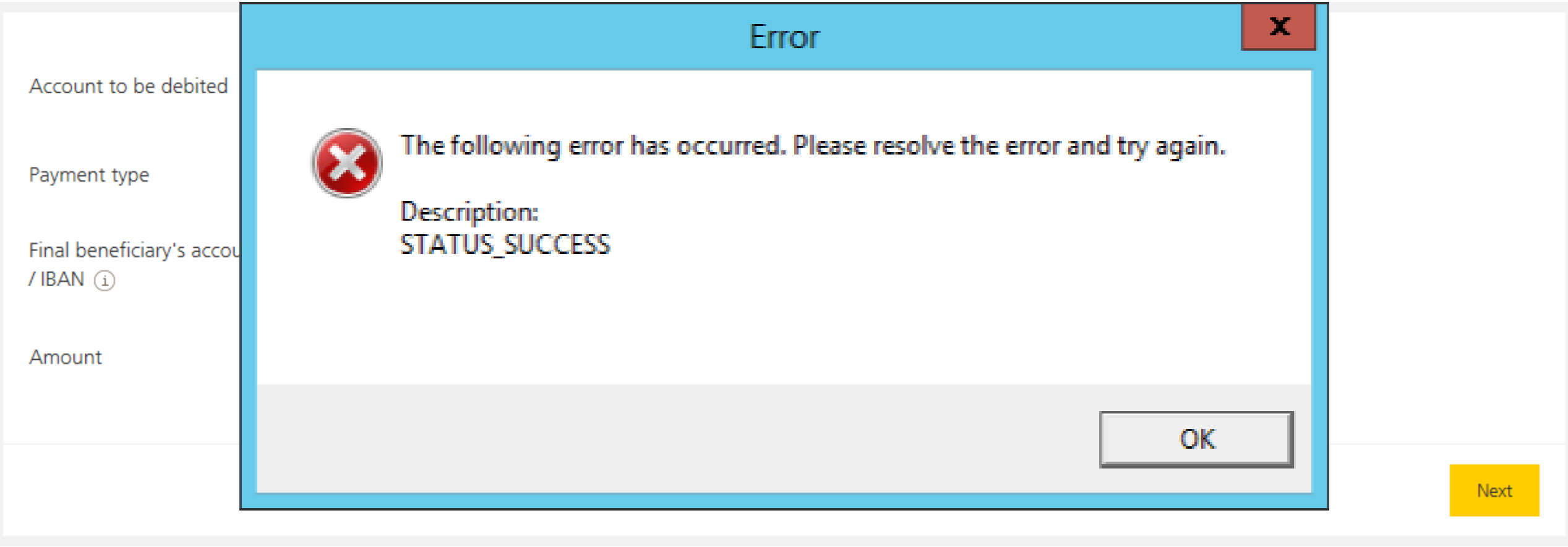
Why?



Federal number:	12822706
SCIPER N°	223572
E-mail:	solal.pirelli@epfl.ch
Registered on:	15.02.2018
Exmatriculation on	

Course						# / max enrolments	Enrolment to exams
							No visible field
Plan EDOC Thesis							
Oral exam							-
Admission EDOC Program courses Doctoral courses Core courses							
CS-724 - Advanced lo and quantum computi						*0 inscript.*	-
COM-702 - Advanced Cryptology						*3 inscript.*	-
ENG-704 - EECS Seminar: Advanced Topics in Machine Learning	Cevher Volkan, Faltings Boi, Frossard Pascal, Jaggi Martin, Sayed Ali H., West Robert	L:28H	2	☐	Every year. Next time: Spring 2020	*0 inscript.*	-
CS-726 - Machine Learning for Database Systems	Ailamaki Anastasia, Koch Christoph	L:14H, Pw:14H	2	☐	Next time: Fall 2019	*11 inscript.*	-
CS-721 - Privacy at the communication layer	González Troncoso Carmela	Re:14H, L:26H	2	☑	Next time: Spring 2018	*6/20*	✓ -

Why?

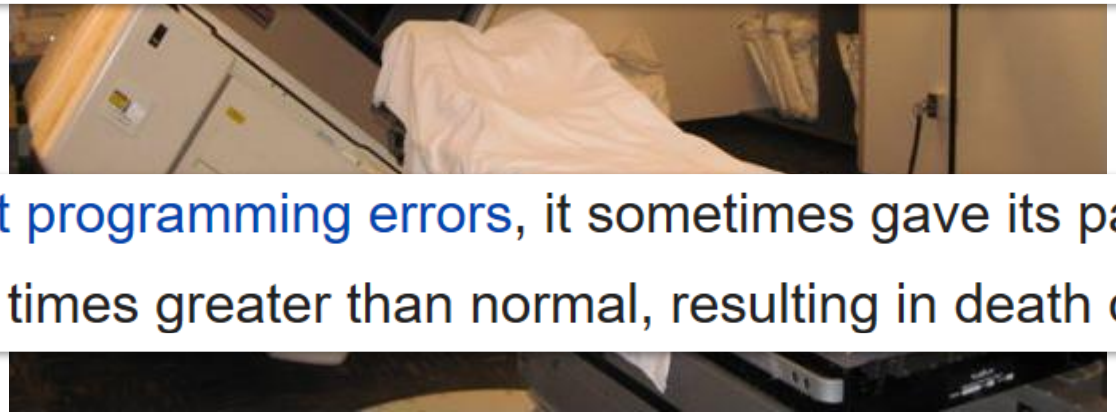

Account to be debited
Payment type
Final beneficiary's accou
/ IBAN
Amount
Error
The following error has occurred. Please resolve the error and try again.
Description:
STATUS_SUCCESS
OK
Next

Why?

Therac-25

From Wikipedia, the free encyclopedia

The **Therac-25** was a computer-controlled radiation therapy machine



Because of concurrent programming errors, it sometimes gave its patients radiation doses that were hundreds of times greater than normal, resulting in death or serious injury.[2]