Gestion des exceptions

Question 1 : Que se passe-t-il en `C++` si une exception est levée mais pas interceptée ?

le programme se termine.

Question 2 : Le gestionnaire d'exceptions fourre-tout C++ s'écrit _____

catch(...)

Question 3 : Le code susceptible de générer une exception doit être écrit dans le bloc _____.

try

Question 4 : Une exception est levée à l'aide du mot-clé _____ ?

throw

Question 5 : La bibliothèque standard `C++` définit un riche ensemble de classes d'exception dérivées de _____.

std::exception

Question 6 : La fonction virtuelle `what()` est d'abord définie en _____.

std::exception

Question 7 : Quel est le résultat du programme suivant ?

```
#include <iostream>
using namespace std;

int main()
{
   try {
       throw 'X';
   }
   catch (const int &ex)
   {
      cout << "Integer exception" << endl;
   }
   catch (...)
   {
      cout << "Any exception" << endl;
   }
   cout << "Goodbye" << endl;
   return 0;
}
```

Any execption

Question 8 : Quel est le résultat du programme suivant ?

```
#include <iostream>
void func_a();
void func_b();
void func_c();

void func_a() {
    std::cout << "Starting func_a" << std::endl;
    func_b();
    throw 3000;
    std::cout << "Ending func_a" << std::endl;
}

void func_b() {
    std::cout << "Starting func_b" << std::endl;
    func_c();
    throw 2000;
    std::cout << "Ending func_b" << std::endl;
}

void func_c()  {
    std::cout << "Starting func_c" << std::endl;
    throw 1000;
    std::cout << "Ending func_c" << std::endl;
}

int main() {
    try {
        func_a();
    } catch(int &ex) {
        std::cerr << "Caught error thrown: " << ex << std::endl;
    }
    std::cout << "Continuing in main" << std::endl;
    return 0;
}
```

sortie :

```
Starting func_a
Starting func_b
Starting func_c
Caught error thrown: 1000
Continuing in main
```

Question 9 : Quel est le problème avec le code suivant ?

```
#include <iostream>
#include <exception>

// this function only knows how to double values
// that are less than 5000

int double_it(int val) {
   if (val >= 5000)
      throw std::exception();
   return val * 2;
 }

int main() {
   try {
      int *values = new int[10000];
      for (int i=0; i< 10000; ++i)
         values[i] = double_it(i); // double_it may throw a std::exception
      delete [] values;

   }
   catch (const std::exception &ex) {
      std::cerr << "Don't know how to double numbers > 5000" << std::endl;
   }

   std::cout << "Program complete" << std::endl;
   return 0;
}
```

il perd la mémoire car `delete[] values` ne s'éxécutera pas

Question 10 : Lequel des énoncés suivants est vrai concernant les classes d'exceptions définies par l'utilisateur ?

*Il peut s'agir de classes C++ classiques *