Module : IDAI-54 Programmation Orientée Objet en C++/Java

Devoir

Botaina Ahadri

IDAI S5

Le nombre de paramètres qu'une fonction peut accepter en C++ n'est pas strictement défini par le langage lui-même, mais il est généralement limité par plusieurs facteurs, notamment :

- **Limites de la pile (Stack Size) :**

- Chaque appel de fonction utilise une certaine quantité de mémoire sur la pile pour stocker les paramètres et d'autres informations d'état. La taille de la pile est limitée (souvent entre 1 Mo et 8 Mo par défaut selon les systèmes). Si vous dépassez cette limite, cela peut entraîner un dépassement de pile (stack overflow).

- **Limites du compilateur :**

- Certains compilateurs peuvent imposer des limites sur le nombre d'arguments. Par exemple, avec GCC, il n'y a pas de limite stricte, mais la capacité de gérer des arguments peut varier en fonction de la mémoire disponible.

- **Mémoire disponible:**

- La mémoire totale disponible sur votre système peut également affecter le nombre de paramètres que vous pouvez passer à une fonction. Si la mémoire est insuffisante, cela peut provoquer des erreurs.

➢ **Exemples de limitations pratiques :**

- Taille de la pile : La pile est limitée en taille, et chaque paramètre occupe de l'espace en mémoire. Pour des entiers (4 octets chacun), 1000 paramètres prendraient environ 4000 octets (4 Ko), ce qui pourrait facilement saturer une pile limitée.

- Compilateurs : Certains compilateurs imposent une limite à la taille de la déclaration d'une fonction. Par exemple, bien que GCC ne limite pas explicitement le nombre de paramètres, l'appel d'une fonction avec un grand nombre de paramètres peut générer une erreur de dépassement de pile.

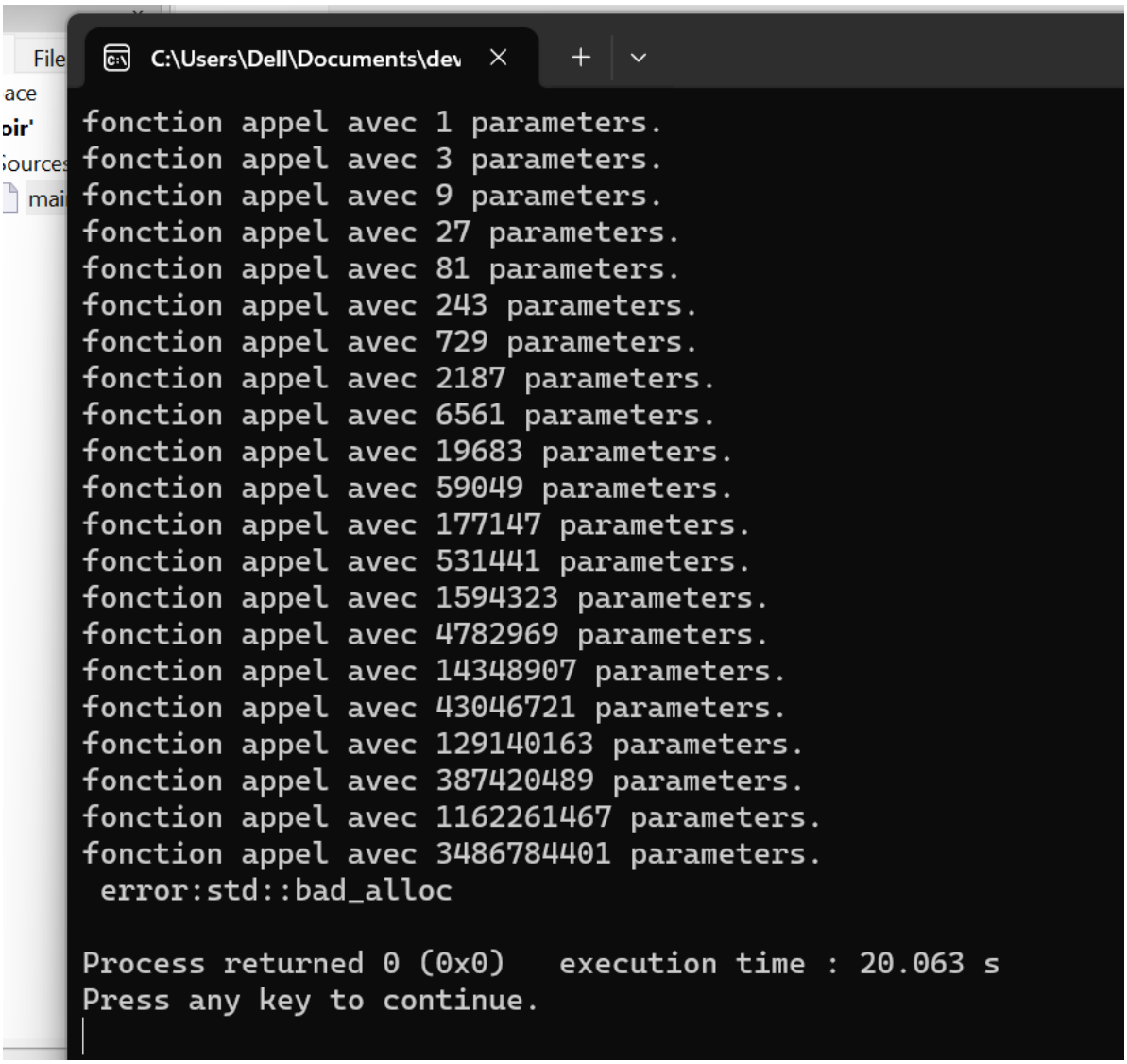
➢ Pour moi , j'ai cherché et trouvé ce code pour trouver les paramètres possibles dans la même fonction :

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

void testfnct( const vector <int>&par){
 cout <<"fonction appel avec "<< par.size()<< " parameters."<<endl;

 }
int main(){
 try {
 for( size_t i=1 ;  ;i*=3){
   vector<int> par(i,1);
   testfnct(par);
 }
 } catch(const bad_alloc& e){
 cout << " error:" << e.what() << endl;
 }
 return 0;
}
```

- Il ma donné :



« Il est impossible de travailler avec ce nombre dans une fonction »