

```

5
6   int main()
7   {
8       std::cout << "Hello World- solution 2!\n";
9   }
10

```

```

5
6   int main() { ... }
10


```

Point d'entrée (entry point)

```

3
4   #include <iostream>
5
6   int main()
7   {
8       std::cout << "Hello World- solution 2!\n";
9   }

```



cout – lié à la console

cout – écrire dans la console

cin – lire de la console

'<<' - inséré dans la console

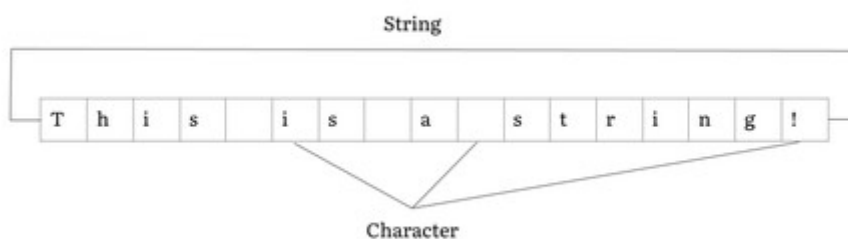
“Enter your favorite number between 1 and 100!” – string

```

dd
1   int main()
2   {
3       std::cout << "Enter your favorite number between 1 and 100!";
4   }

```

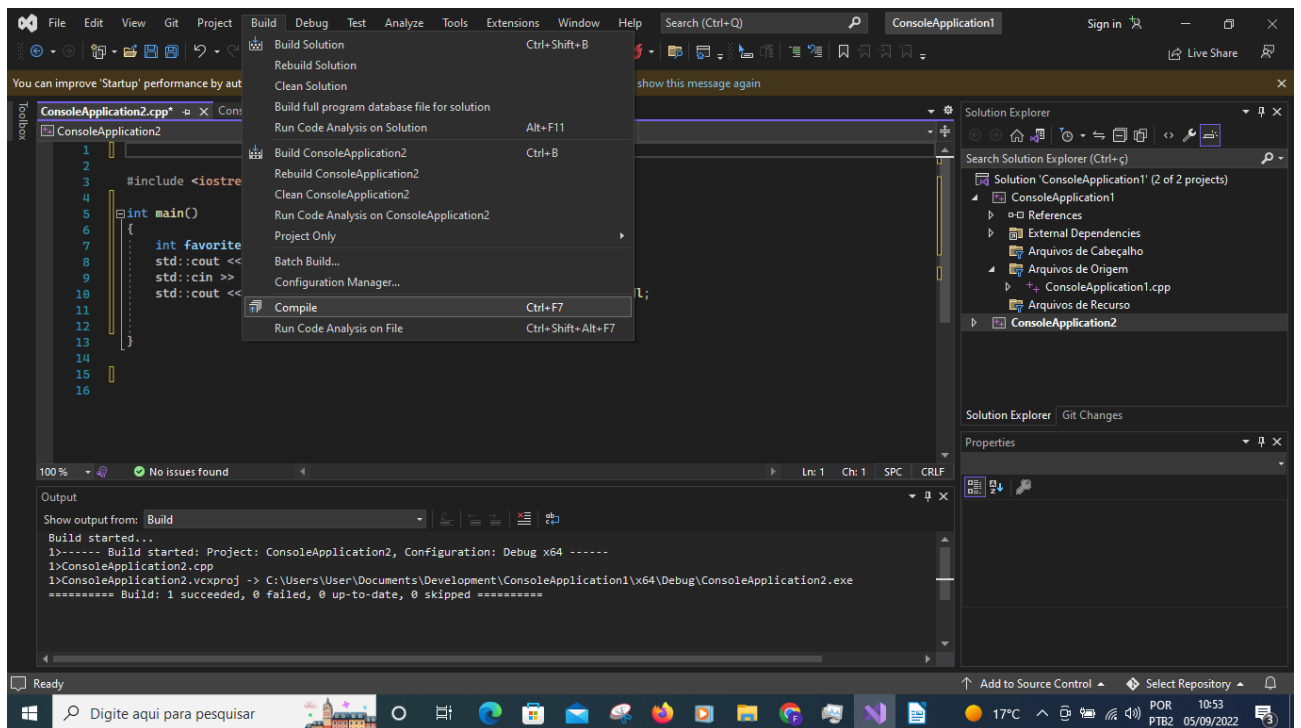
- String- séquence des caractères:



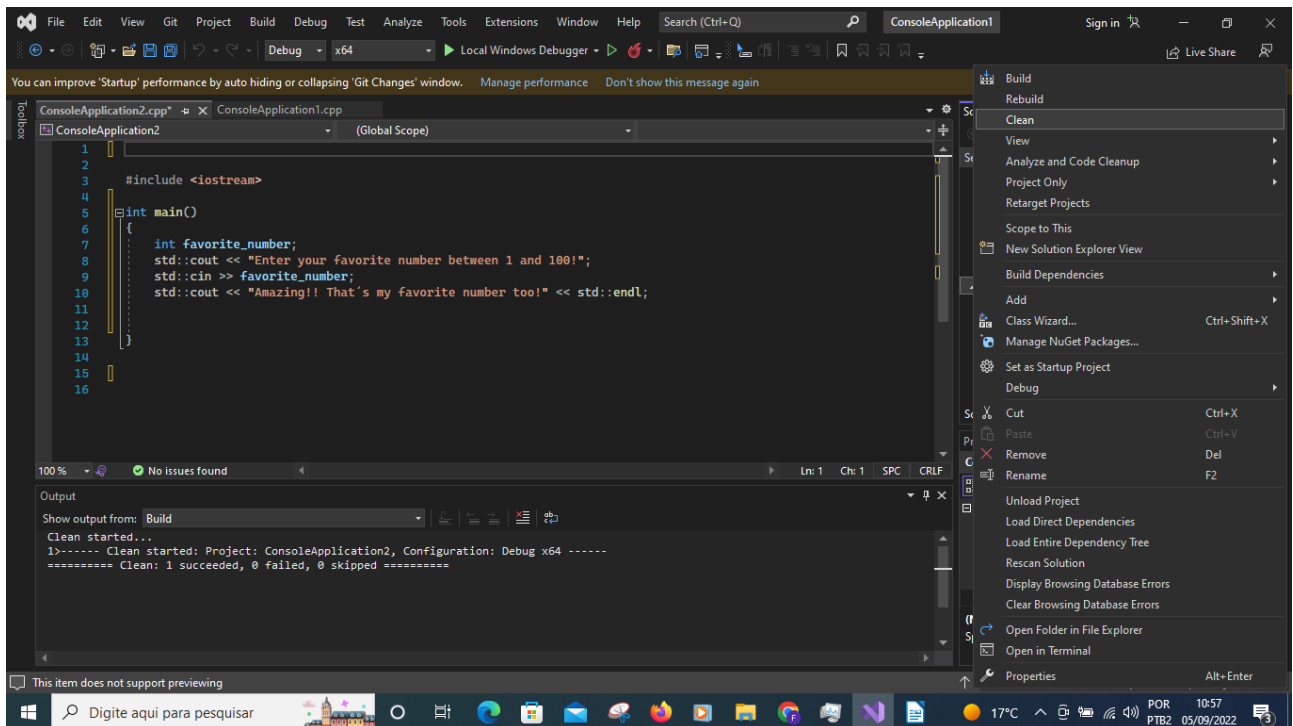
Include <iostream>:

Pour utiliser input et output (cout et cin).

```
#include <iostream>
```



Clean:



Rebuild = clean + build.

Erreurs:

Le compilateur cheche les erreurs.

Syntax errors:

```
3
4 #include <iostream>
5
6 int main()
7 {
8     int favorite_number;
9     std::cout << "Enter your favorite number between 1 and 100 !";
10    std::cin >> favorite_number;
11    std::cout << "Amazing!! That's my favorite number too!" << std::endl;
12
13    std::cout << "hi";
14
15    return 0;
16 }
```

0% 2 0 Ln: 9 Ch: 6

or List

ntire Solution 4 Errors 0 Warnings 0 of 1 Message Build + IntelliSense Search Error List

| | Code | Description | Project | File | Line | Suppress |
|-----|-------|---|---------------------|-------------------------|------|----------|
| abs | E0008 | missing closing quote | ConsoleApplication1 | ConsoleApplication1.cpp | 9 | |
| abs | E0065 | expected a ';' | ConsoleApplication1 | ConsoleApplication1.cpp | 10 | |
| ✖ | C2001 | newline in constant | ConsoleApplication1 | ConsoleApplication1.cpp | 9 | |
| ✖ | C2143 | syntax error: missing ';' before 'std::cin' | ConsoleApplication1 | ConsoleApplication1.cpp | 10 | |

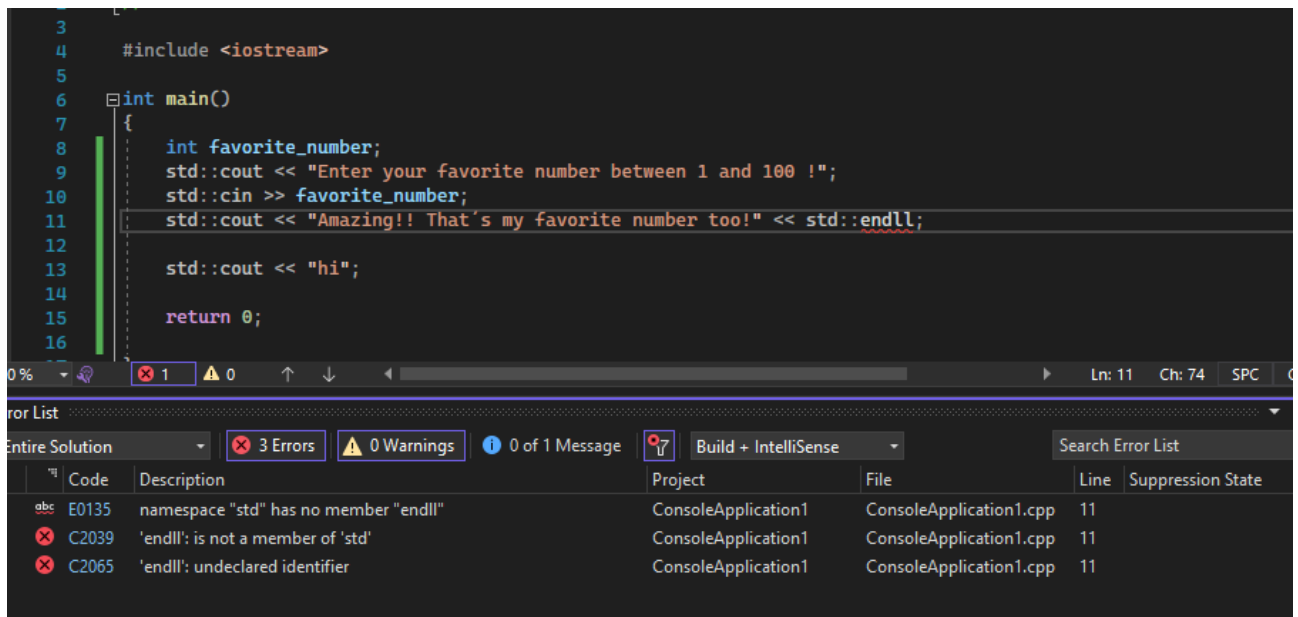
```
1 // ConsoleApplication1.cpp : Este arquivo contém a função 'main'. A execução do programa começa e termina
2 //
3
4 #include <iostream>
5
6 int main()
7 {
8     int favorite_number;
9     std::cout << "Enter your favorite number between 1 and 100 !";
10    std::cin >> favorite_number;
11    std::cout << "Amazing!! That's my favorite number too!" << std::endl;
12
13    std::cout << "hi";
14
15    return 0
16 }
17
18
19
20
```

0% 1 0 Ln: 15 Ch: 13 SPC

or List

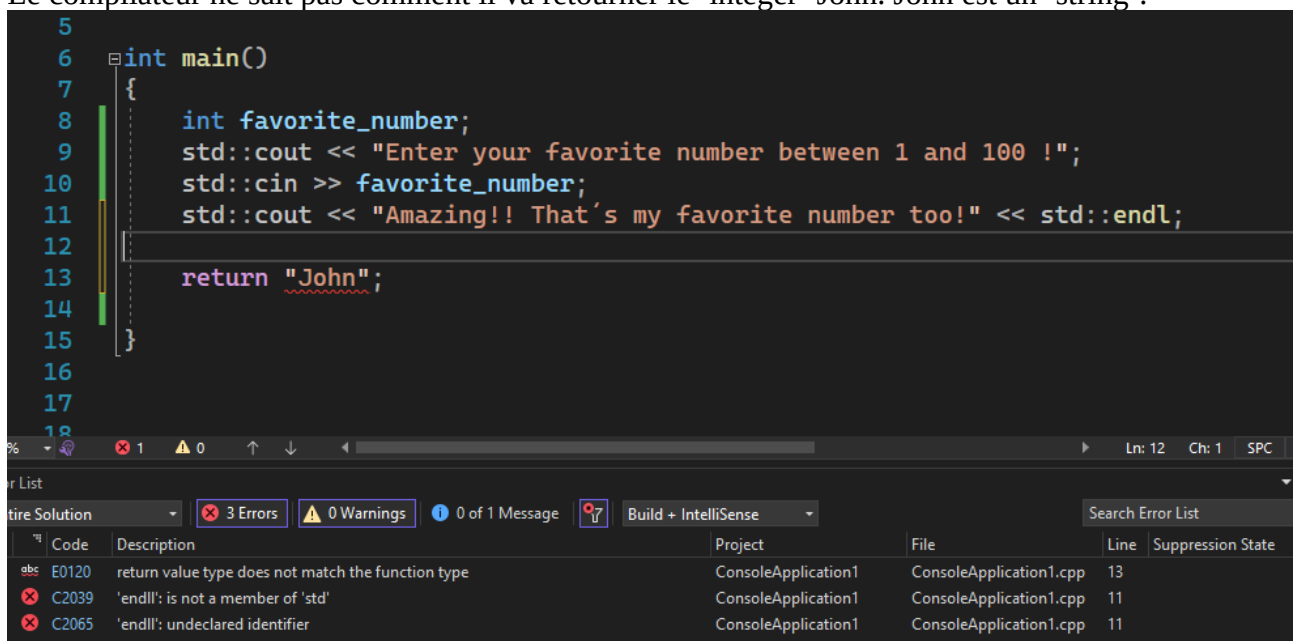
ntire Solution 2 Errors 0 Warnings 0 Messages Build + IntelliSense Search Error List

| | Code | Description | Project | File | Line | Suppression State |
|-----|-------|---------------------------------------|---------------------|-------------------------|------|-------------------|
| abs | E0065 | expected a ';' | ConsoleApplication1 | ConsoleApplication1.cpp | 17 | |
| ✖ | C2143 | syntax error: missing ';' before ';'' | ConsoleApplication1 | ConsoleApplication1.cpp | 17 | |

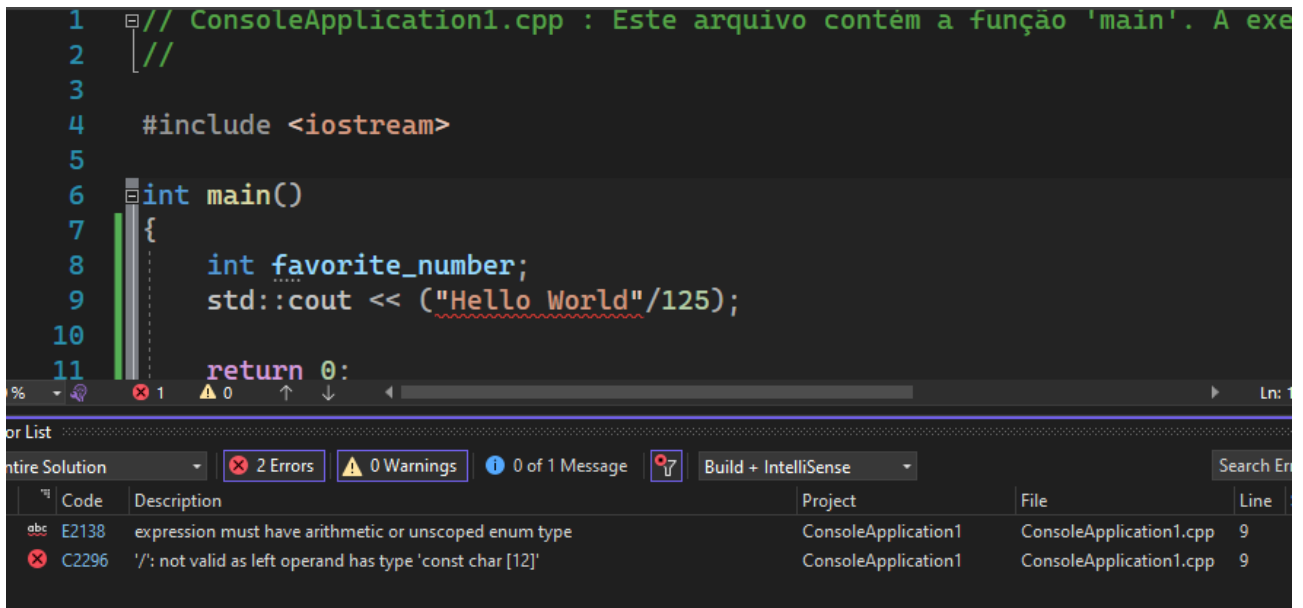


Semantic errors:

Le compilateur ne sait pas comment il va retourner le 'integer' John. John est un 'string'.



Un autre type d'erreur de syntaxe:



```
1 // ConsoleApplication1.cpp : Este arquivo contém a função 'main'. A exe
2 //
3
4 #include <iostream>
5
6 int main()
7 {
8     int favorite_number;
9     std::cout << ("Hello World"/125);
10
11     return 0;
12 }
```

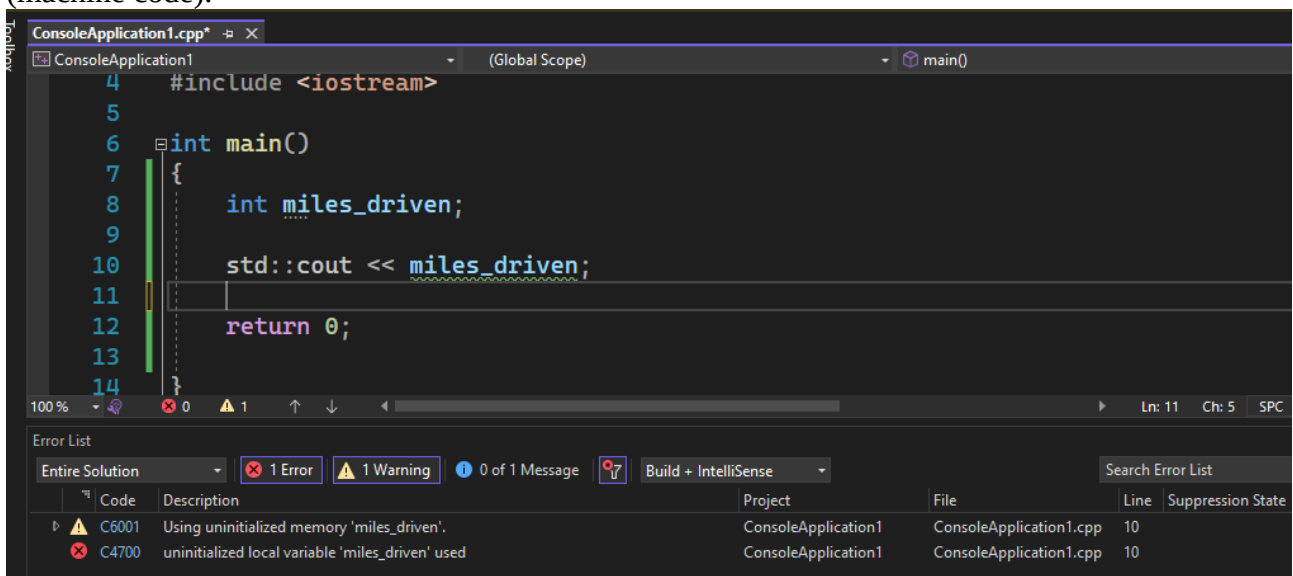
2 Errors 0 Warnings 0 of 1 Message Build + IntelliSense

| Code | Description | Project | File | Line |
|-------|---|---------------------|-------------------------|------|
| E2138 | expression must have arithmetic or unscoped enum type | ConsoleApplication1 | ConsoleApplication1.cpp | 9 |
| C2296 | '/': not valid as left operand has type 'const char [12]' | ConsoleApplication1 | ConsoleApplication1.cpp | 9 |

Compile Warnings:

Le compilateur reconnaît un problème dans le code qui peut apporter une erreur.

Par contre, c'est juste un avertissement (warning) parce que le compilateur peut créer le code (machine code).



```
4 #include <iostream>
5
6 int main()
7 {
8     int miles_driven;
9
10     std::cout << miles_driven;
11
12     return 0;
13 }
14 }
```

1 Error 1 Warning 0 of 1 Message Build + IntelliSense

| Code | Description | Project | File | Line | Suppression State |
|-------|--|---------------------|-------------------------|------|-------------------|
| C6001 | Using uninitialized memory 'miles_driven'. | ConsoleApplication1 | ConsoleApplication1.cpp | 10 | |
| C4700 | uninitialized local variable 'miles_driven' used | ConsoleApplication1 | ConsoleApplication1.cpp | 10 | |

ConsoleApplication1 (Global Scope) main()

```
4 #include <iostream>
5
6 int main()
7 {
8     int favorite_number;
9
10    std::cout << "Hello, world";
11
12    return 0;
13 }
14
15
16
17
```

No issues found

Error List

Entire Solution 0 Errors 1 Warning 1 Message Build + IntelliSense Search Error List

| Code | Description | Project | File | Line |
|------------|--|---------------------|-------------------------|------|
| Int-uninit | Local variable is not initialized. | ConsoleApplication1 | ConsoleApplication1.cpp | 8 |
| C4101 | 'favorite_number': unreferenced local variable | ConsoleApplication1 | ConsoleApplication1.cpp | 8 |

ConsoleApplication1 (Global Scope) main()

```
1 // ConsoleApplication1.cpp : Este arquivo contém a função 'main'. A execução do programa começa e termina ali.
2 //
3
4 #include <iostream>
5
6 extern int x;
7
8 int main()
9 {
10    int favorite_number;
11
12    std::cout << "Hello, world";
13
14    std::cout << x;
15
16    return 0;
17 }
18
19
```

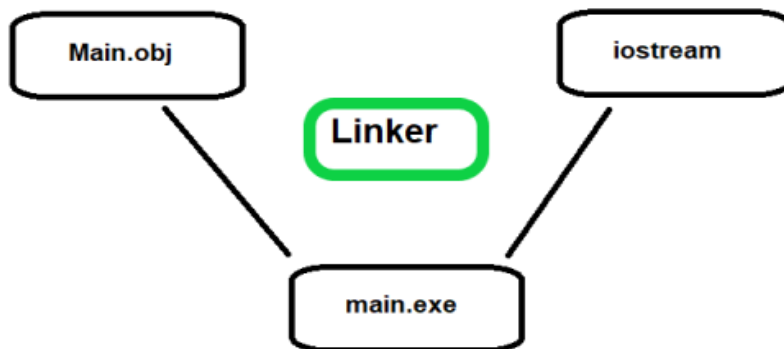
100 % No issues found

Error List

Entire Solution 2 Errors 0 Warnings 1 Message Build + IntelliSense Search Error List

| Code | Description | Project | File | Line | Suppression State |
|------------|---|---------------------|-------------------------|------|-------------------|
| Int-uninit | Local variable is not initialized. | ConsoleApplication1 | ConsoleApplication1.cpp | 10 | |
| LNK2001 | unresolved external symbol "int x" (?x@@@3HA) | ConsoleApplication1 | ConsoleApplication1.obj | 1 | |
| LNK1120 | 1 unresolved externals | ConsoleApplication1 | ConsoleApplication1.exe | 1 | |

Linker



Runtime errors:

Quand on roule le programme:

Division par zéro, fichier non trouvé, 'out of memory'.

Exception Handling peut aider.

Logic errors:

```
[?]  
#include <iostream>  
  
extern int x;  
  
int main()  
{  
    int age;  
    if (age > 18) {  
        std::cout << "yes, you can vote!";  
    }  
    return 0;  
}
```

Exercice:

Écrivez un programme qui demande le numéro favori. Après, lisez le numéro dans la console. Après, vérifiez si le numéro est plus grand ou égal à 22. Montrez la réponse dans la console.

Défi:

Vérifiez si le numéro est situé entre 22(inclusivement) et 54(exclusivement). Imprimez la réponse.

Solution:

```
#include <iostream>

int main()
{
    int age;
    std::cin >> age;
    if (age >= 22) {
        if (age < 54) {
            std::cout << "il est situé entre 22(inclusivement) et 54(exclusivement)";
        }
        else {
            std::cout << "Hors de la plage 22-54";
        }
    }

    return 0;
}
```


Structure:

Keywords: les mots clés réservés en C++. Ils ne peuvent pas être redéfinis.

C++ = 90

Java = 50

C = 32

Python = 33

<https://en.cppreference.com/w/cpp/keyword>

cppreference.com

Create account

Search

Page

Discussion

View

Edit

History

C++

C++ language

Keywords

C++ keywords

This is a list of reserved keywords in C++. Since they are used by the language, these keywords are not available for re-definition or overloading.

| A - C | D - P | R - Z |
|------------------------|------------------|---------------------------|
| alignas (C++11) | decltype (C++11) | constexpr (reflection TS) |
| alignof (C++11) | default (1) | register (2) |
| and | delete (1) | reinterpret_cast |
| and_eq | do | requires (C++20) |
| asm | double | return |
| atomic_cancel (TMTS) | dynamic_cast | short |
| atomic_commit (TMTS) | else | signed |
| atomic_noexcept (TMTS) | enum | sizeof (1) |
| auto (1) | explicit | static |
| bitand | export (1) (3) | static_assert (C++11) |
| bitor | extern (1) | static_cast |
| bool | false | struct (1) |
| break | float | switch |
| case | for | synchronized (TMTS) |
| catch | friend | template |
| char | goto | this (4) |
| char8_t (C++20) | if | thread_local (C++11) |
| char16_t (C++11) | inline (1) | throw |
| char32_t (C++11) | int | true |
| class (1) | long | try |
| compl | mutable (1) | typedef |
| concept (C++20) | namespace | typeid |
| const | new | typename |
| constexpr (C++20) | noexcept (C++11) | union |

Identifieurs:

Identifieurs != Keywords

Identifieurs- le programmeur donne les noms.

Opérateurs:

“> >”, “< <”, “{”, “;”

Syntax:

La structure et le contenu que vous voulez que le compilateur comprenne.

Preprocessor directives:

C'est logiciel que traite votre code source avant le compilateur. Il enlève les commentaires. Après, il va chercher le "#".

Il remplace la ligne avec le "#" et insère le fichier auquel la ligne fait référence.

Il peut être utilisé pour vérifier le SO (système opérationnel):

```
#ifdef _WIN32
// do something for windows like include <windows.h>
#elif defined __unix__
// do something for unix like include <unistd.h>
#elif defined __APPLE__
// do something for mac
#endif
```

```
#include <iostream>
#include "myfile.h"
#if
#elif
```

<https://en.cppreference.com/w/cpp/preprocessor>

Commentaires:

Description de ce qu'il fait, et pas comment il fait.

Exemple:

-une ligne

//This is a comment

-plusieurs lignes

**/* This is a multiple line
comment */**

Le compilateur va ignorer le commentaire.

```
//=====
//***** FUNCTIONS *****
//=====
```

Ne pas faire ça, utiliser git (github, gitlab), pour contrôler la version:

```

/*****
* author Paul

*10/10/2017 – Fixed bug in ...
*05/05/2019 – Added function ...
*****/

```

Modifier le code et le commentaire aussi.

La fonction main():

Chaque programme doit avoir 1(une) fonction “main()”.

Quand on excute un programme, la fonction main () est apellé et le code à l’interieur des acollades est executé.

Quand l’exécution atteint la ligne “return 0”, le programme retourne le integer au système opérationnel.

Deux types de main ().

1-

```

int main( ) {
    //code
    return 0;
}

```

2-

```

int main (int argc, char *argv[ ]) {
    //code
    return 0;
}

```

Namespaces:

Pour éviter les conflits entre les bibliothèques.

Namespace= containeur pour garder l

```

#include <iostream>
#include <string>

namespace utilz {
    void imprimer(std::string nom){
        std::cout << "hello, " + nom;
    }
}

int main()
{
    utilz::imprimer("john");

    return 0;
}

```

```

using namespace std;

#include <iostream>
using namespace std;

namespace utilz {
    void imprimer(string nom){
        std::cout << "hello, " + nom;
    }
}

int main()
{
    utilz::imprimer("john");

    return 0;
}

```

Une autre façon d'utiliser le namespace:

```

#include <iostream>
using std::cout;
using std::cin;
using std::string;

namespace utilz {
    void imprimer(string nom){
        std::cout << "hello, " + nom;
    }
}

int main()
{
    utilz::imprimer("john");

    return 0;
}

```

Basic I/O avec cin et cout:

-inséré des données: `cout << data;`

-enchaîner les données: `cout << "data 1 is: " << data1`

-le ligne n'est pas sauté automatiquement:

```

cout << "data 1 is: " << data1 << endl;
cout << "data 1 is: " << data1 << "\n";

```

cin et >>:

```
cin >> data;
```

pour enchaîner:

```
cin >> data1 >> data2;
```

==

```
#include <iostream>
```

```

using namespace std;

int main()
{
    cout << "Hello, world" << endl;

    cout << "Hello";
    cout << "Hello" << endl;

    cout << "Hello, world" << endl;
    cout << "Hello" << "world" << endl;
    cout << "Hello, world\n";
    cout << "Hello\nOut\nThere\n";

    int num1;
    int num2;
    double num3;

    cout << "enter an integer";
    cin >> num1;
    cout << "You entered: " << num1 << endl;

    cout << "enter a first integer";
    cin >> num1;

    cout << "enter a second integer";
    cin >> num2;

    cout << "You entered" << num1 << "and" << num2 << endl;

    cout << "enter 2 integers separated with a space";
    cin >> num1 >> num2;
    cout << "You entered" << num1 << " and " << num2 << endl;

    return 0;
}

```

Exercice

Créer un namespace avec une fonction pour imprimer un numéro int et un numéro double.
Ex. Les numéros sont 12 et 10.5.

Les numéros doivent être lus à partir de la console à partir d'une seule ligne.