Is it fine to write “void main()” or “main()” in C/C++?

The definition

```
void main() { /* ... */ }
```

is not and never has been C++, nor has it even been C. See the ISO C++ standard 3.6.1[2] or the ISO C standard 5.1.2.2.1. A conforming implementation accepts

```
int main() { /* ... */ }
```

and

```
int main(int argc, char* argv[]) { /* ... */ }
```

A conforming implementation may provide more versions of main(), but they must all have return type int. The int returned by main() is a way for a program to return a value to “the system” that invokes it. On systems that doesn’t provide such a facility the return value is ignored, but that doesn’t make “void main()” legal C++ or legal C. ***Even if your compiler accepts “void main()” avoid it, or risk being considered ignorant by C and C++ programmers. In C++, main() need not contain an explicit return statement. In that case, the value returned is 0, meaning successful execution.*** For example:

```
#include <iostream>
int main()
{
   std::cout << "This program returns the integer value 0\n";
}
```

Note also that neither ISO C++ nor C99 allows you to leave the type out of a declaration. That is, in contrast to C89 and ARM C++ ,”int” is not assumed where a type is missing in a declaration. Consequently:

```
#include <iostream>

main() { /* ... */ }
```

Run on IDE

is an error because the return type of main() is missing.

Source: http://www.stroustrup.com/bs_faq2.html#void-main

To summarize above, it is never a good idea to use “void main()” or just “main()” as it doesn’t confirm standards. It may be allowed by some compilers though.