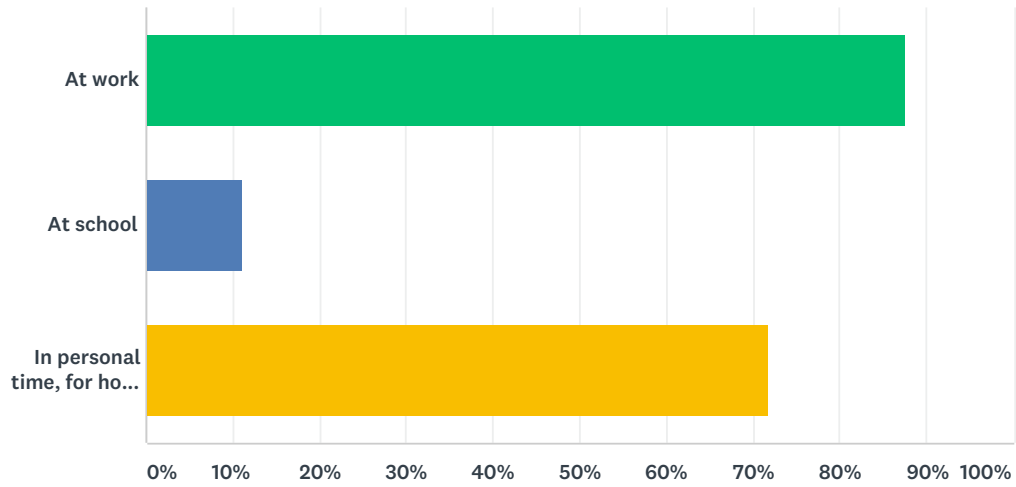


## Q1 Where do you use C++? (select all that apply)

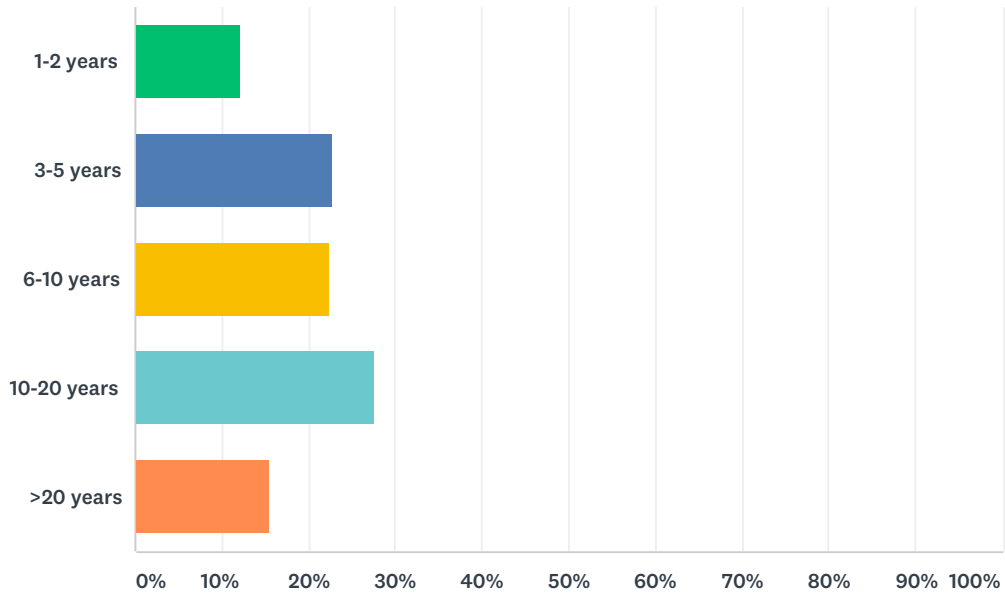
Answered: 2,154 Skipped: 2



ANSWER CHOICES	RESPONSES	
At work	87.70%	1,889
At school	11.10%	239
In personal time, for hobby projects or to try new things	71.73%	1,545
Total Respondents: 2,154		

## Q2 How many years of programming experience do you have in C++ specifically?

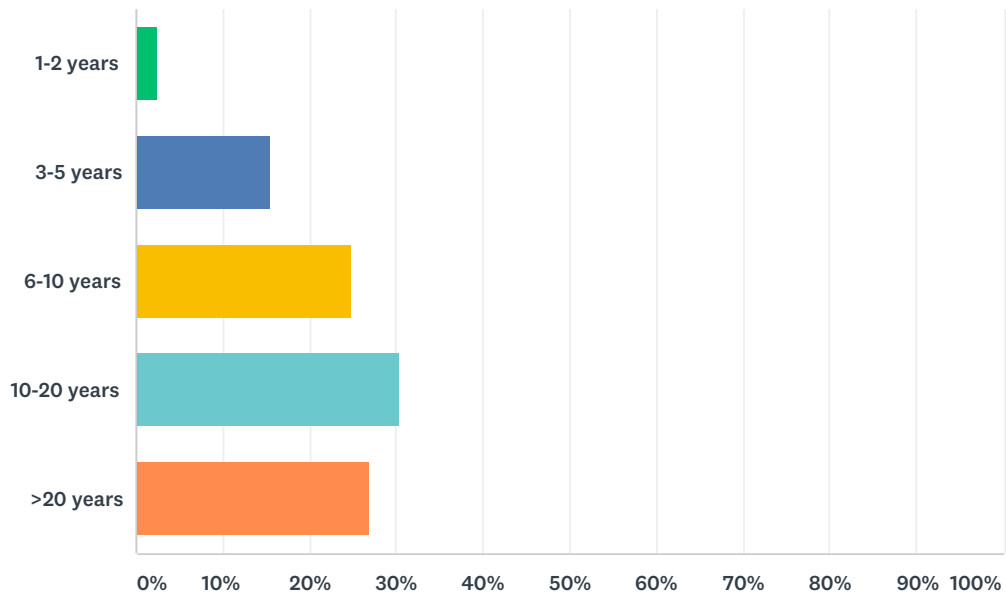
Answered: 2,151 Skipped: 5



ANSWER CHOICES	RESPONSES	
1-2 years	12.04%	259
3-5 years	22.73%	489
6-10 years	22.32%	480
10-20 years	27.52%	592
>20 years	15.39%	331
TOTAL		2,151

### Q3 How many years of programming experience do you have overall (all languages)?

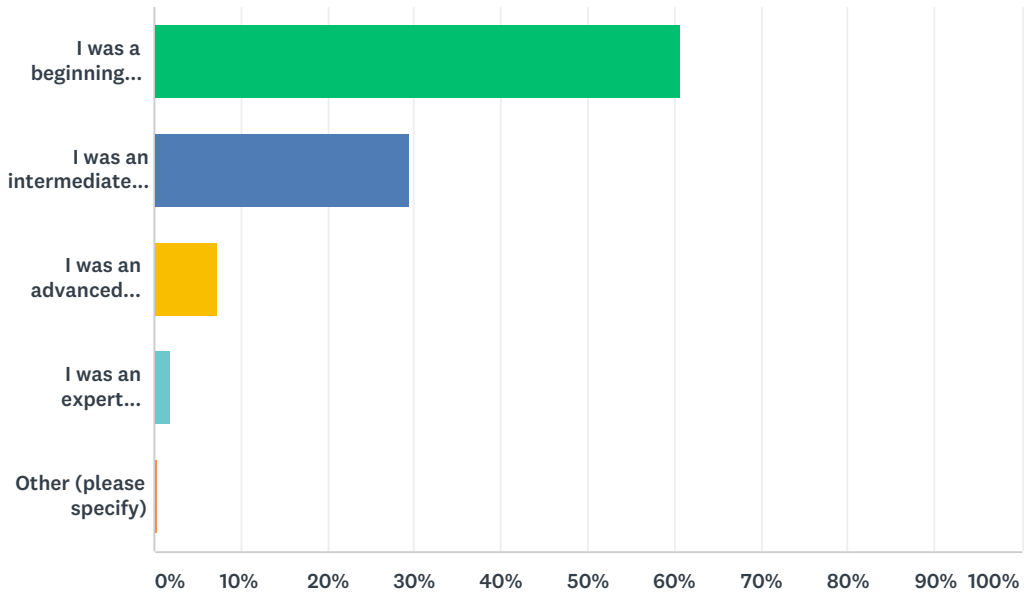
Answered: 2,153 Skipped: 3



ANSWER CHOICES	RESPONSES	
1-2 years	2.46%	53
3-5 years	15.47%	333
6-10 years	24.90%	536
10-20 years	30.28%	652
>20 years	26.89%	579
TOTAL		2,153

## Q4 When you first learned C++, how much programming experience did you already have?

Answered: 2,154 Skipped: 2



ANSWER CHOICES	RESPONSES	
I was a beginning programmer	60.77%	1,309
I was an intermediate programmer	29.57%	637
I was an advanced programmer	7.34%	158
I was an expert programmer	1.81%	39
Other (please specify)	0.51%	11
<b>TOTAL</b>		<b>2,154</b>

## Q5 What did you find to be the most challenging aspects of learning C++?

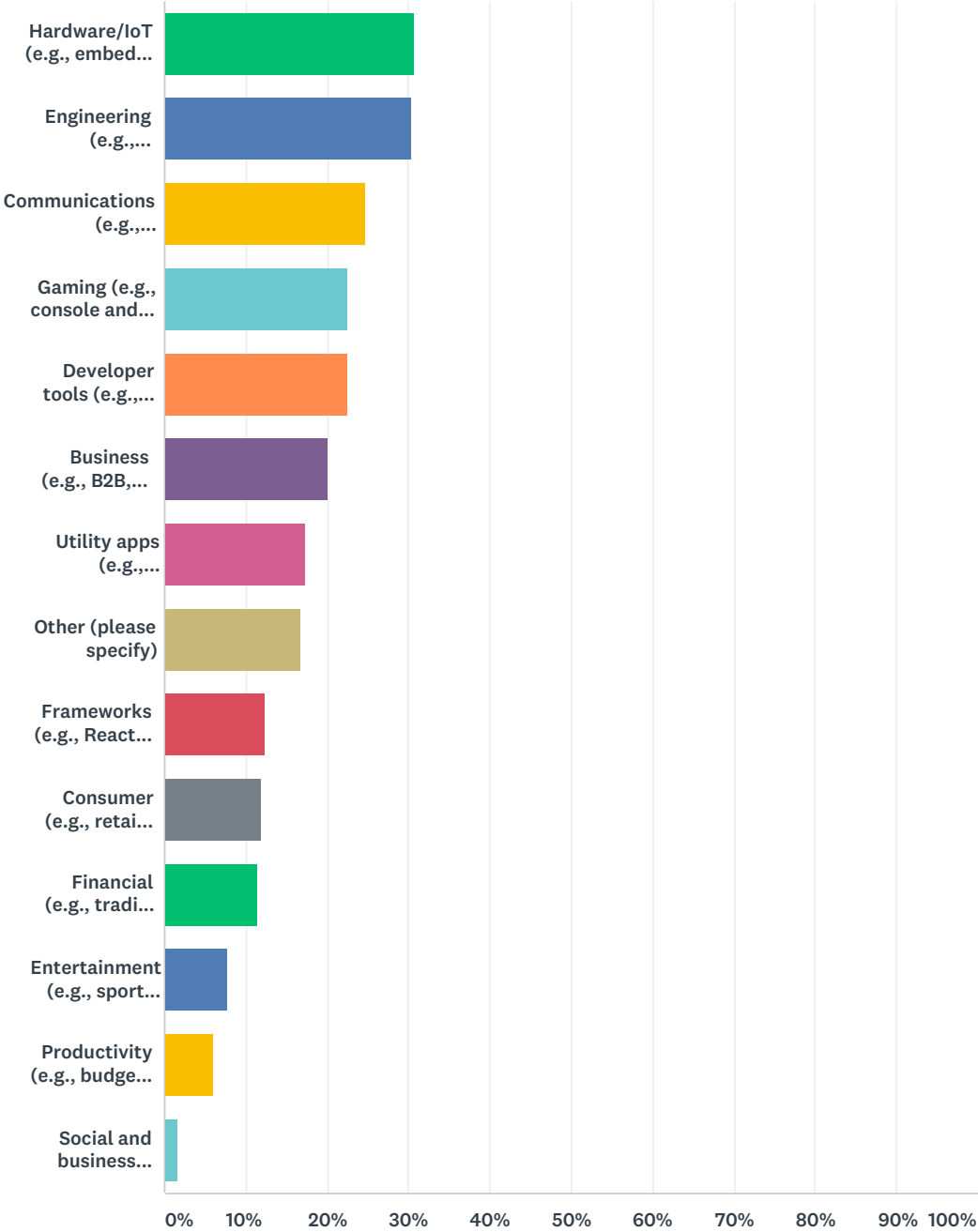
Answered: 1,559 Skipped: 597

### Q5 What did you find to be the most challenging aspects of learning C++?

A word cloud visualization showing the most challenging aspects of learning C++ based on 1,559 responses. The words are arranged in a roughly rectangular shape, with larger words indicating higher frequency. The most prominent words include 'challenging', 'template', 'meta-programming', 'learning', 'C', 'classes', 'best practices', 'specific', 'much', 'new features', 'think', 'modern', 'classes', 'difficult', 'references', 'really', 'way', 'stuff', 'concepts', 'problem', 'etc', 'types', 'Lack', 'later', 'different', 'ways', 'thing', 'move', 'semantics', 'new', 'error', 'messages', 'bad', 'code', 'make', 'build', 'systems', 'easy', 'syntax', 'beginner', 'compiler', 'need', 'programming', 'undefined', 'behavior', 'time', 'standard', 'libraries', 'one', 'understand', 'complex', 'learning', 'project', 'language', 'modern C', 'C', 'STL', 'templates', 'Pointers', 'references', 'pointers', 'books', 'memory', 'management', 'now', 'use', 'first', 'template', 'metaprogramming', 'Pointers', 'memory', 'things', 'back', 'good', 'write', 'hard', 'e.g', 'work', 'Debugging', 'find', 'OOP', 'meta', 'programming', 'complexity', 'lot', 'part', 'tooling', 'RAII', 'metaprogramming', 'Template', 'error', 'many', 'resources', 'remember', 'SFINAE', 'still', 'details', 'features', 'read', 'know', 'inheritance', 'standard', 'library', 'documentation', 'rules', 'smart pointers', 'template', 'meta', 'large', 'Also', 'many ways', 'started', 'compiler', 'linking', 'error', 'build'.

### Q6 What types of projects do you work on? (select all that apply)

Answered: 2,140 Skipped: 16



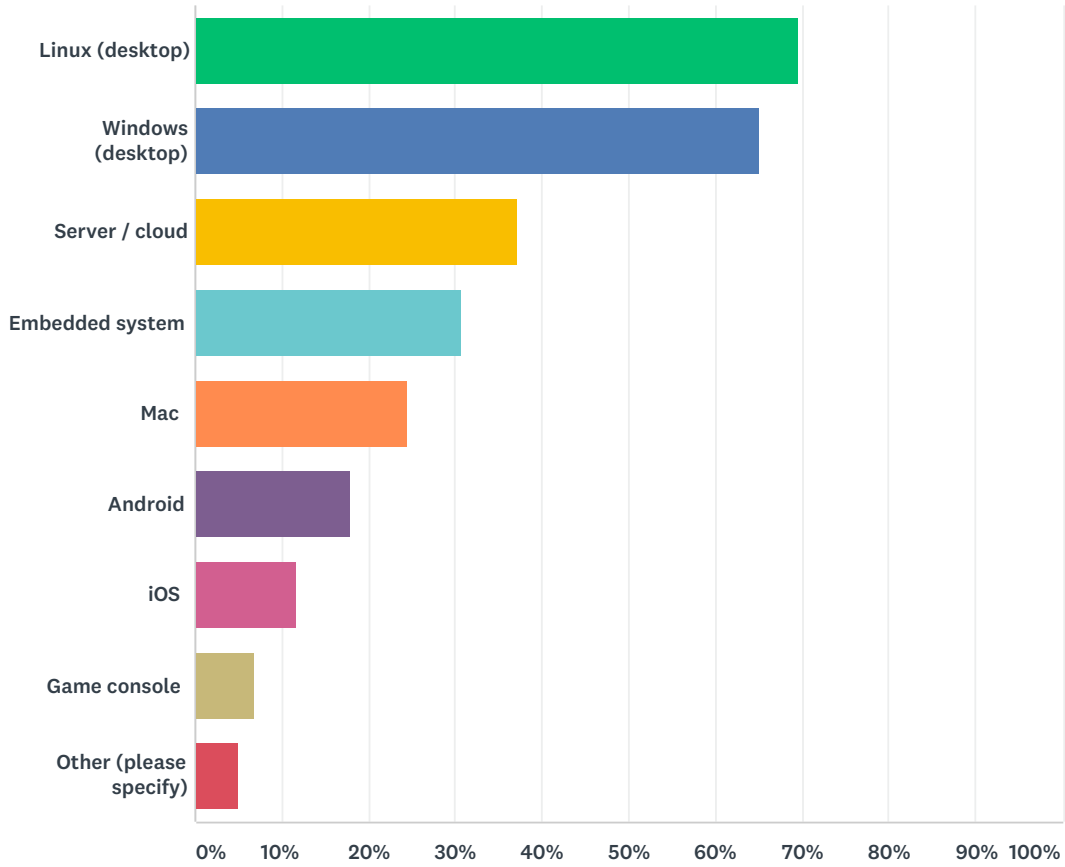
ANSWER CHOICES	RESPONSES	
Hardware/IoT (e.g., embedded systems, home automation)	30.70%	657
Engineering (e.g., avionics, power management)	30.33%	649
Communications (e.g., networking, email)	24.72%	529
Gaming (e.g., console and mobile games)	22.57%	483
Developer tools (e.g., compilers, code editors)	22.52%	482

## 2019 Annual C++ Developer Survey "Lite"

Business (e.g., B2B, B2E)	20.05%	429
Utility apps (e.g., calculators, simple image editors)	17.34%	371
Other (please specify)	16.64%	356
Frameworks (e.g., React, Unity)	12.43%	266
Consumer (e.g., retail websites, mobile apps)	11.96%	256
Financial (e.g., trading, mortgage, asset management)	11.45%	245
Entertainment (e.g., sports apps, video streaming)	7.66%	164
Productivity (e.g., budget tracking, note taking)	6.17%	132
Social and business networking (e.g., Facebook, Twitter)	1.78%	38
Total Respondents: 2,140		

## Q7 What platforms do you develop for? (select all that apply)

Answered: 2,147 Skipped: 9

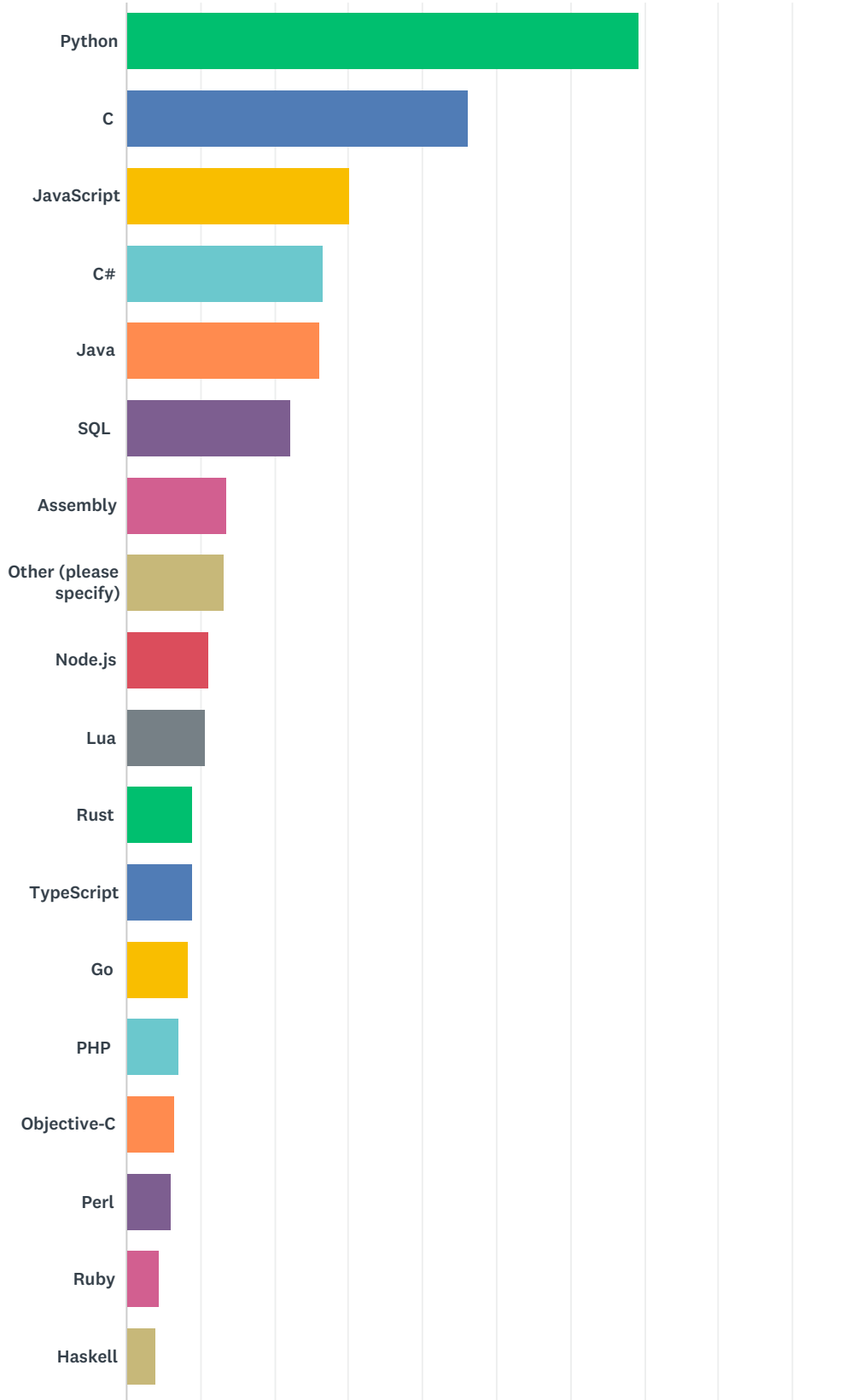


ANSWER CHOICES	RESPONSES	
Linux (desktop)	69.63%	1,495
Windows (desktop)	65.11%	1,398
Server / cloud	37.17%	798
Embedded system	30.79%	661
Mac	24.45%	525
Android	17.93%	385
iOS	11.69%	251
Game console	6.94%	149
Other (please specify)	5.08%	109
Total Respondents: 2,147		

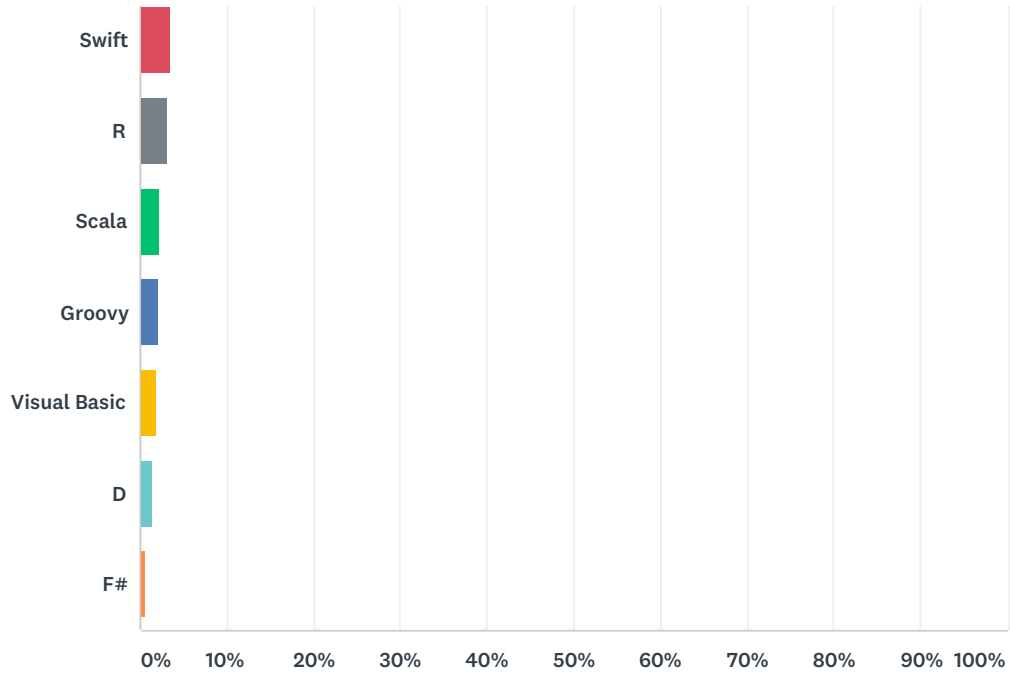


# Q8 Besides C++, what programming languages/environments do you use in your current and recent projects? (select all that apply)

Answered: 2,073 Skipped: 83



## 2019 Annual C++ Developer Survey "Lite"



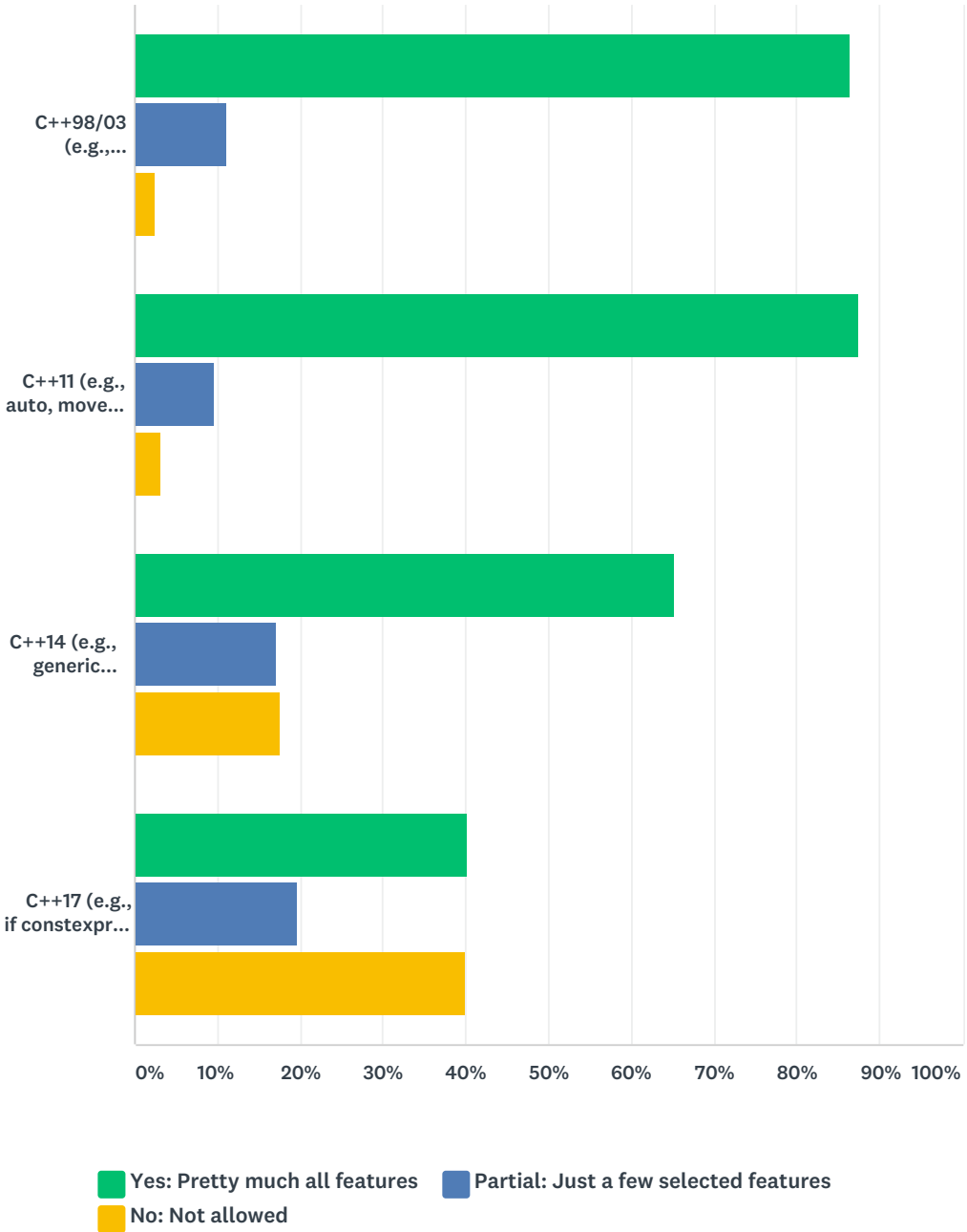
ANSWER CHOICES	RESPONSES	
Python	69.18%	1,434
C	46.26%	959
JavaScript	30.20%	626
C#	26.63%	552
Java	26.19%	543
SQL	22.14%	459
Assembly	13.51%	280
Other (please specify)	13.12%	272
Node.js	11.05%	229
Lua	10.66%	221
Rust	9.07%	188
TypeScript	8.97%	186
Go	8.35%	173
PHP	7.14%	148
Objective-C	6.42%	133
Perl	5.98%	124
Ruby	4.34%	90
Haskell	3.96%	82
Swift	3.52%	73
R	3.04%	63

## 2019 Annual C++ Developer Survey "Lite"

Scala	2.22%	46
Groovy	2.17%	45
Visual Basic	1.98%	41
D	1.40%	29
F#	0.72%	15
Total Respondents: 2,073		

# Q9 What version(s) of C++ are you allowed to use on your current project (work or school)?

Answered: 2,134 Skipped: 22



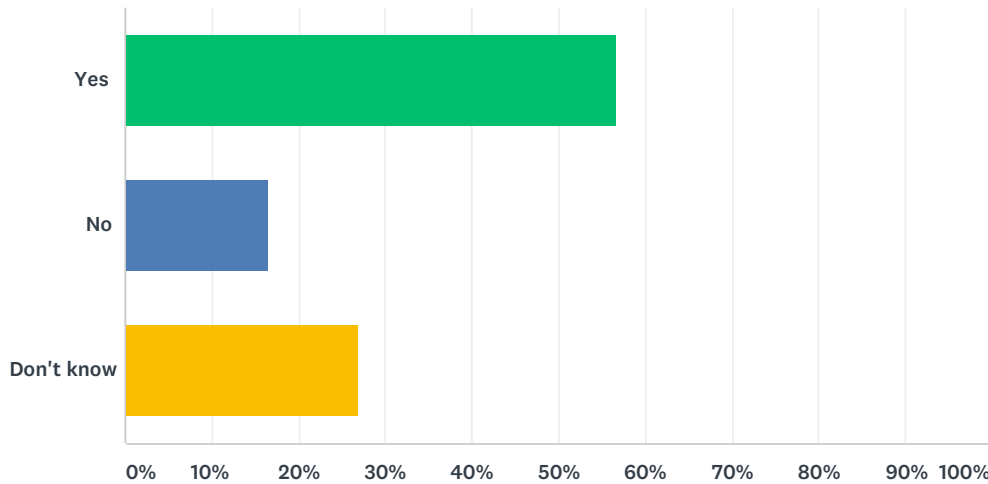
	YES: PRETTY MUCH ALL FEATURES	PARTIAL: JUST A FEW SELECTED FEATURES	NO: NOT ALLOWED	TOTAL	WEIGHTED AVERAGE
C++98/03 (e.g., exceptions, templates, RTTI)	86.39% 1,663	11.01% 212	2.60% 50	1,925	2.84
C++11 (e.g., auto, move semantics, =delete/=default, shared_ptr, lambdas)	87.39% 1,746	9.56% 191	3.05% 61	1,998	2.84

## 2019 Annual C++ Developer Survey "Lite"

C++14 (e.g., generic lambdas, auto return types, general constexpr functions)	65.23% 1,302	17.13% 342	17.64% 352	1,996	2.48
C++17 (e.g., if constexpr, if/switch scoped variables, structured bindings, string_view, optional/any/variant, Parallel STL)	40.25% 815	19.75% 400	40.00% 810	2,025	2.00

### Q10 In the next 12 months, does your current project plan to start allowing additional use of newer C++ standard features (i.e., more than in the previous answer)?

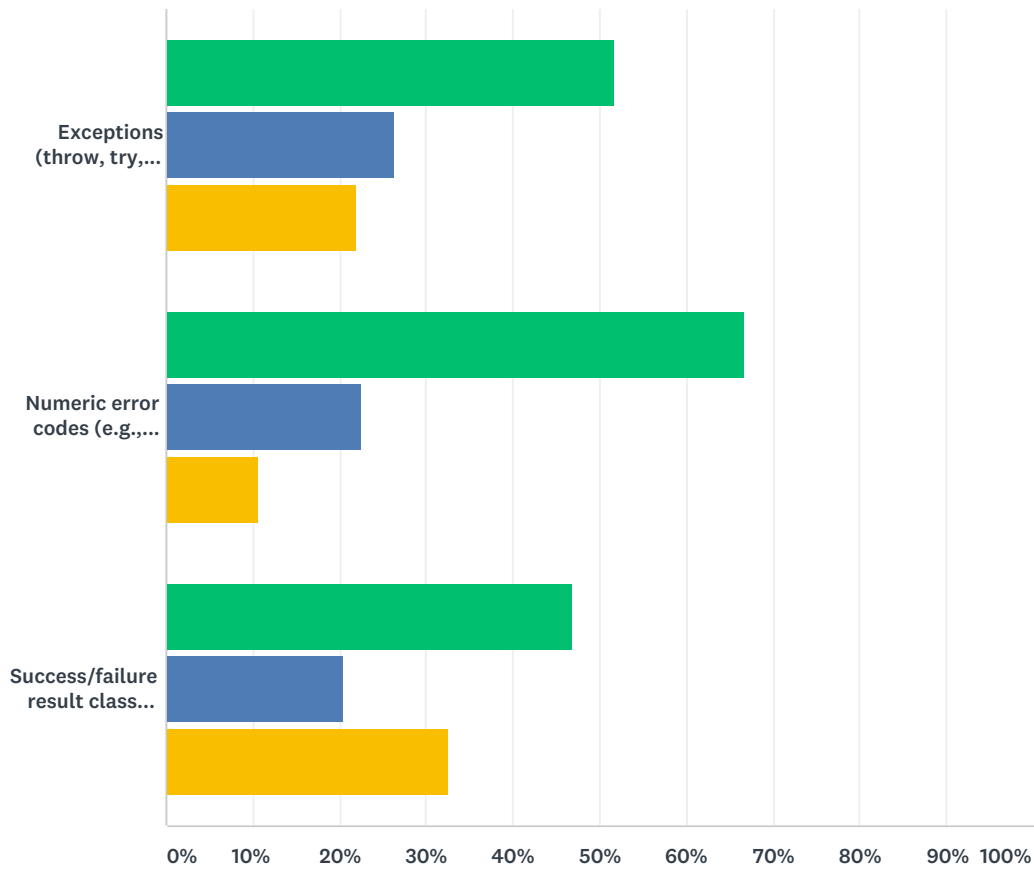
Answered: 2,128 Skipped: 28



ANSWER CHOICES	RESPONSES	
Yes	56.63%	1,205
No	16.45%	350
Don't know	26.93%	573
TOTAL		2,128

# Q11 What error reporting methods are allowed on your current project (work or school)?

Answered: 2,123 Skipped: 33

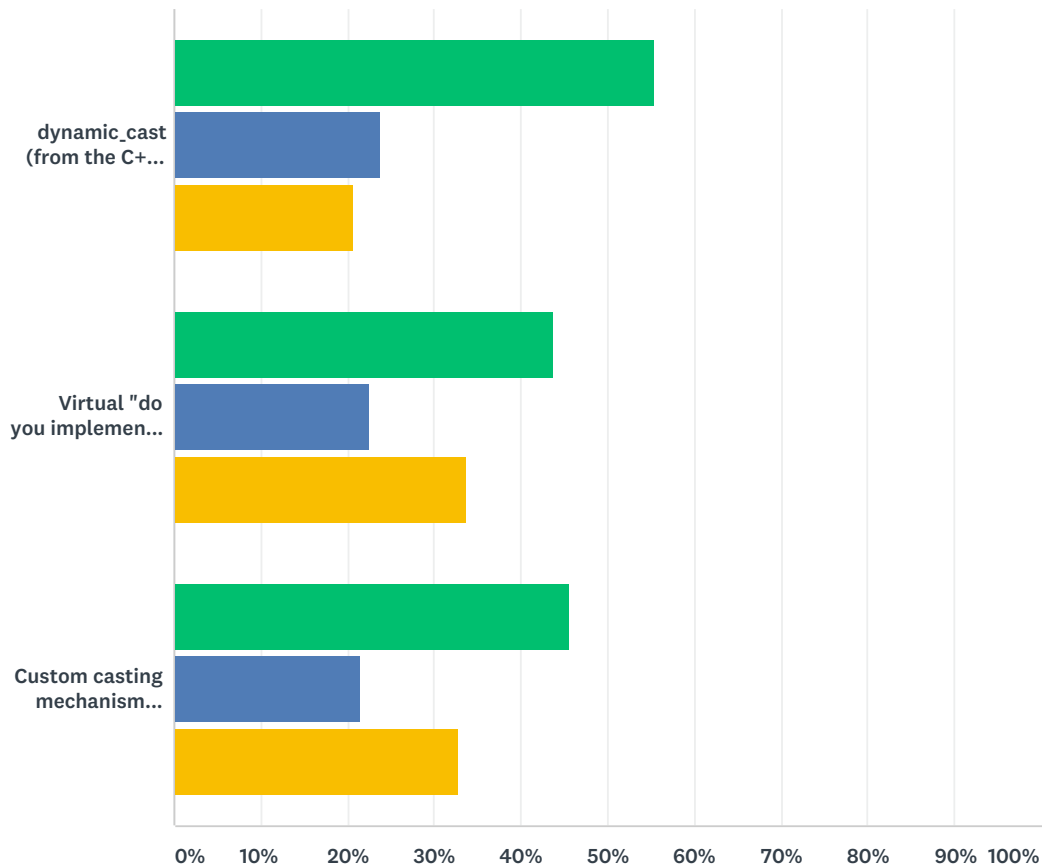


■ Yes: Allowed pretty much anywhere  
■ Partial: Allowed in some parts of the code but not others ■ No: Not allowed

	YES: ALLOWED PRETTY MUCH ANYWHERE	PARTIAL: ALLOWED IN SOME PARTS OF THE CODE BUT NOT OTHERS	NO: NOT ALLOWED	TOTAL	WEIGHTED AVERAGE
Exceptions (throw, try, catch)	51.66% 1,090	26.40% 557	21.94% 463	2,110	2.30
Numeric error codes (e.g., errc, error_code, HRESULT)	66.83% 1,378	22.60% 466	10.57% 218	2,062	2.56
Success/failure result class types (e.g., Boost.Expected, Boost.Outcome)	46.78% 936	20.59% 412	32.63% 653	2,001	2.14

## Q12 What run-time casting methods are allowed on your current project (work or school)?

Answered: 2,083 Skipped: 73



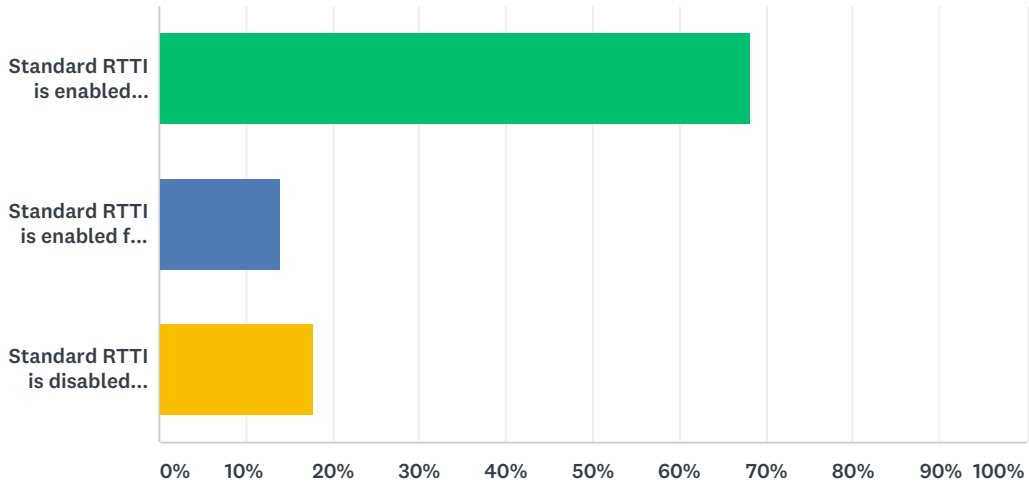
■ Yes: Allowed pretty much anywhere  
■ Partial: Allowed in some parts of the code but not others ■ No: Not allowed

	YES: ALLOWED PRETTY MUCH ANYWHERE	PARTIAL: ALLOWED IN SOME PARTS OF THE CODE BUT NOT OTHERS	NO: NOT ALLOWED	TOTAL	WEIGHTED AVERAGE
dynamic_cast (from the C++ standard)	55.35% 1,149	23.89% 496	20.76% 431	2,076	2.35
Virtual "do you implement" query function (e.g., QueryInterface)	43.64% 857	22.66% 445	33.71% 662	1,964	2.10
Custom casting mechanism (e.g., project-specific type tag and query)	45.59% 890	21.52% 420	32.89% 642	1,952	2.13



Q13 C++ compilers commonly provide a switch, such as `-fno-rtti-data` or `/GR-`, to turn off support for Standard C++ RTTI (e.g., `typeid`, `dynamic_cast`). When building your current project, is standard RTTI enabled, or is it disabled using such a switch?

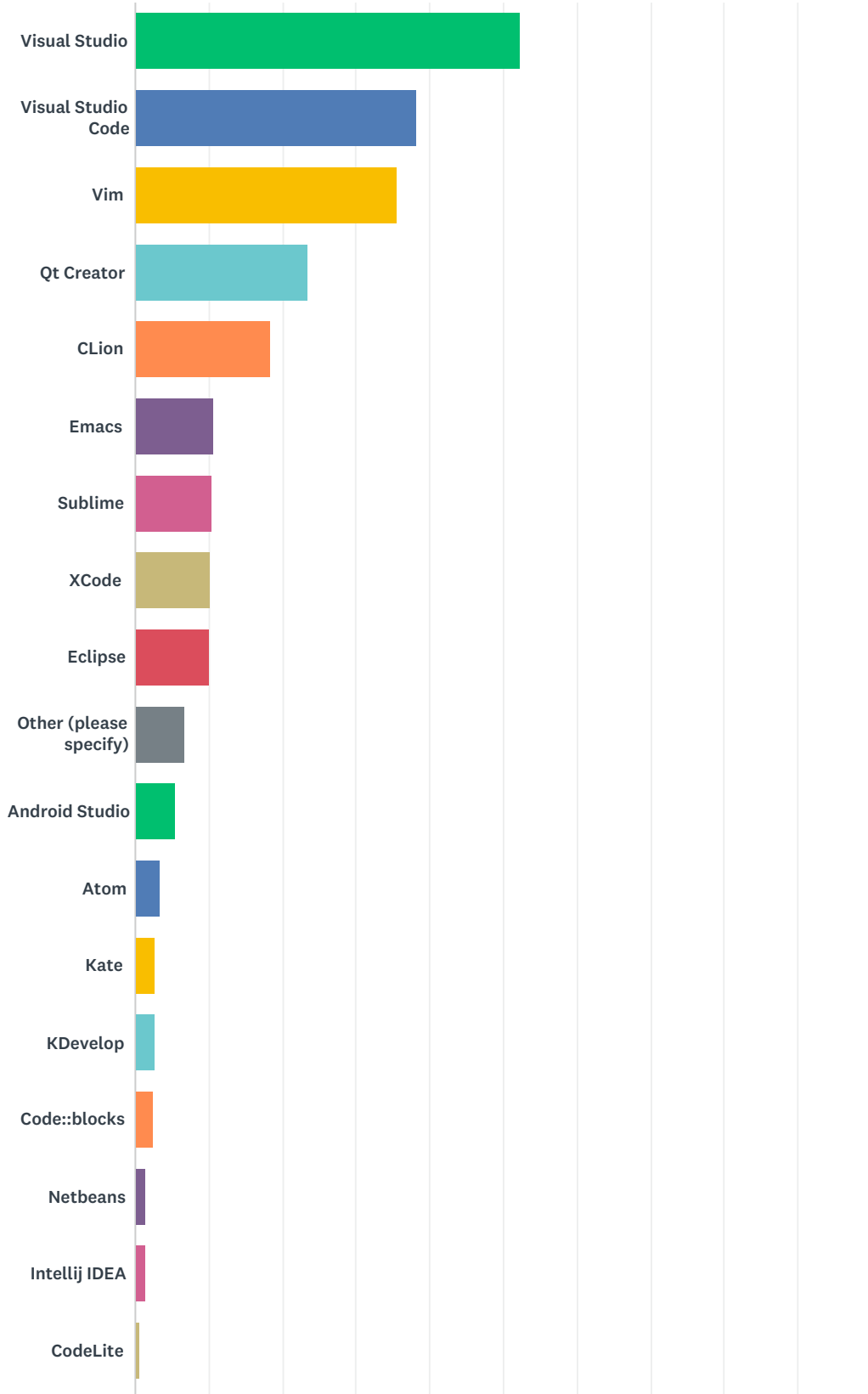
Answered: 2,058 Skipped: 98



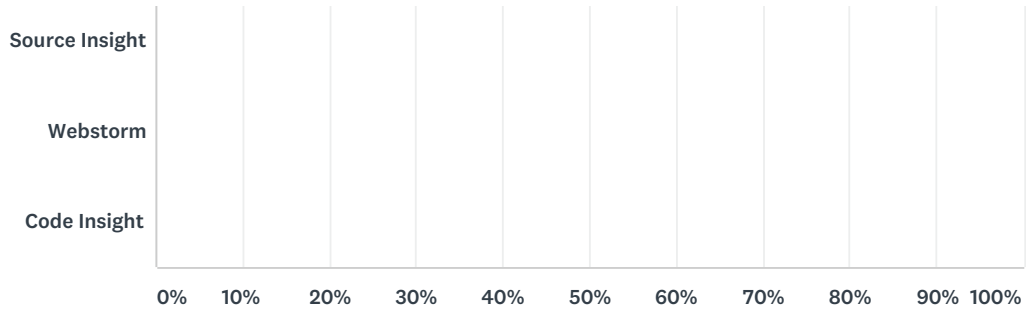
ANSWER CHOICES	RESPONSES	
Standard RTTI is enabled everywhere in my project	68.22%	1,404
Standard RTTI is enabled for building some parts, disabled for building others	14.09%	290
Standard RTTI is disabled everywhere in my project	17.69%	364
TOTAL		2,058

# Q14 Which development environments (IDEs) or editors do you use for C++ development?

Answered: 2,140 Skipped: 16



## 2019 Annual C++ Developer Survey "Lite"



ANSWER CHOICES	RESPONSES	
Visual Studio	52.29%	1,119
Visual Studio Code	38.22%	818
Vim	35.47%	759
Qt Creator	23.46%	502
CLion	18.50%	396
Emacs	10.70%	229
Sublime	10.51%	225
XCode	10.28%	220
Eclipse	10.00%	214
Other (please specify)	6.73%	144
Android Studio	5.42%	116
Atom	3.27%	70
Kate	2.76%	59
KDevelop	2.71%	58
Code::blocks	2.57%	55
Netbeans	1.50%	32
Intellij IDEA	1.40%	30
CodeLite	0.61%	13
Source Insight	0.28%	6
Webstorm	0.23%	5
Code Insight	0.00%	0
Total Respondents: 2,140		

Q15 If you could wave a magic wand and change one thing about any part of C++, what would it be, and how would that change help your daily work?

Answered: 1,404 Skipped: 752

Q15 If you could wave a magic wand and change one thing about any part of C++, what would it be, and how would that change help your daily work?

file string nice check Rust see default standard package simple require  
exceptions even new improve features networking way many allow Fix  
without e.g tools will standard library macros build replace  
compiler debugging remove include change simplify  
easier make easier work standardized modules name  
package manager think standard much  
better implemented code error messages C instead  
make constexpr use less language system  
library also add etc types every compile time  
dependency management package management something  
need error build system backward compatibility things old  
reflection interface project example support implicit conversions  
template Python time parts function know one currently class std  
syntax problem help give lot handling write reduce dependency build times  
const default want

## Q16 Do you have any additional feedback for C++ standardization?

Answered: 803 Skipped: 1,353

### Q16 Do you have any additional feedback for C++ standardization?

coming feel maybe allow nice something see without simple great work since started  
people everything example well one trying new features projects std now  
also every much improve going years great big Thank e.g  
Keep changes new networking modules direction good tools  
things lot work old make compile time use focus  
language hard C stuff features remove  
standard many Please even need move library  
concepts Keep good work easy think love time less  
code know really development way effort will take standardized  
learn add slow reflection committee standard library syntax compiler  
want etc possible write STL support make c type stop exceptions ranges  
standardization great job seems package manager