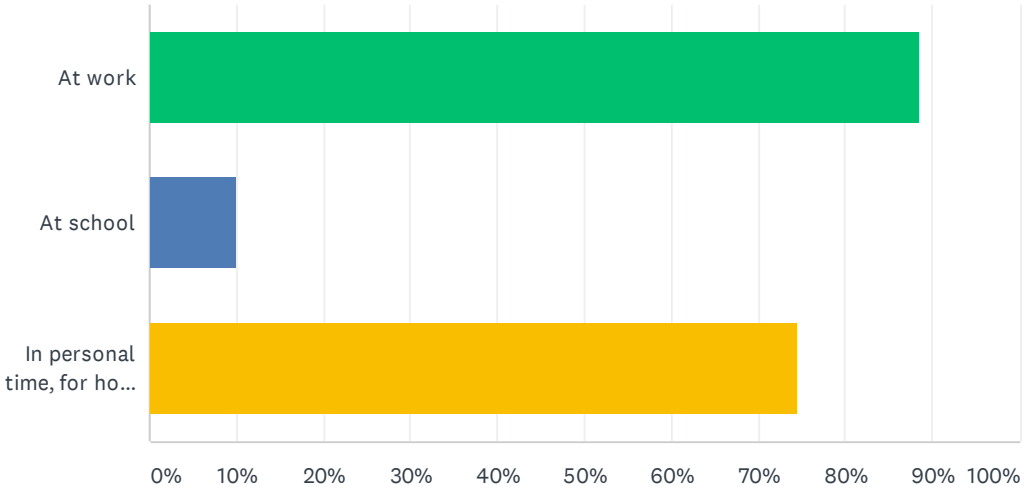
Q1 Where do you use C++? (select all that apply)

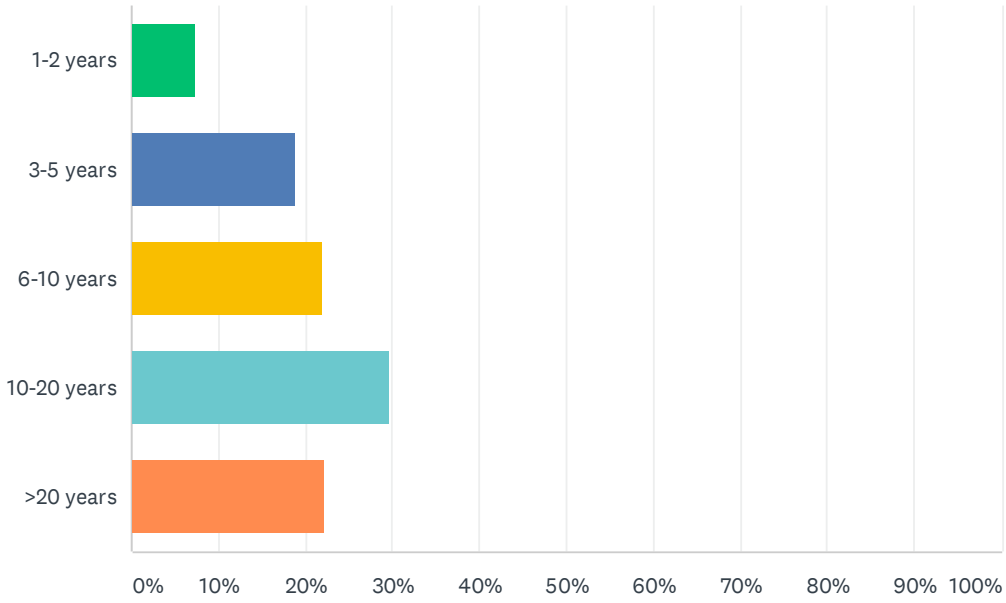
Answered: 1,034 Skipped: 1



ANSWER CHOICES	RESPONSES	
At work	88.39%	914
At school	10.06%	104
In personal time, for hobby projects or to try new things	74.47%	770
Total Respondents: 1,034		

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

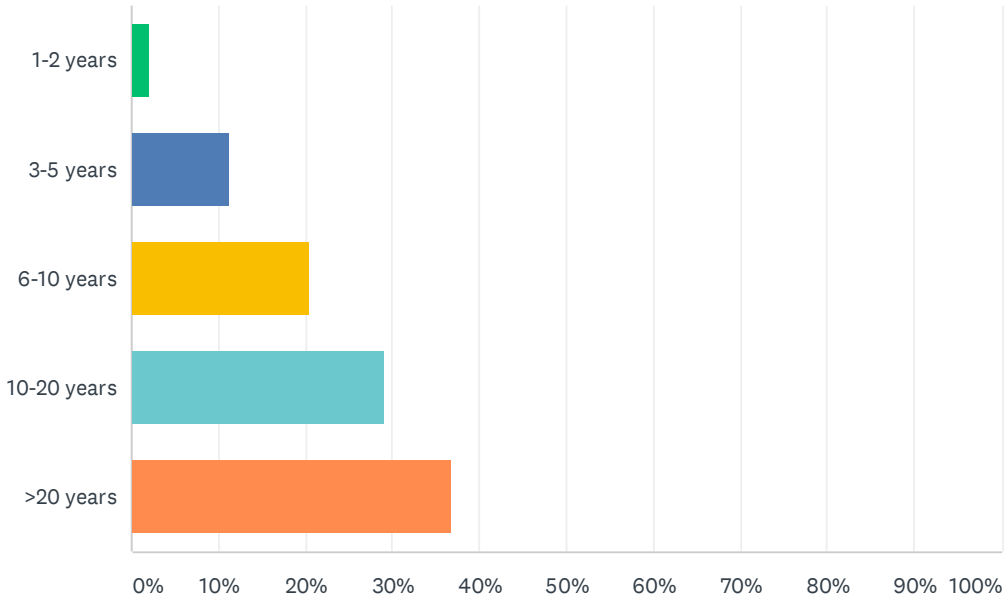
Answered: 1,030 Skipped: 5



ANSWER CHOICES	RESPONSES	
1-2 years	7.28%	75
3-5 years	18.83%	194
6-10 years	21.94%	226
10-20 years	29.81%	307
>20 years	22.14%	228
TOTAL		1,030

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

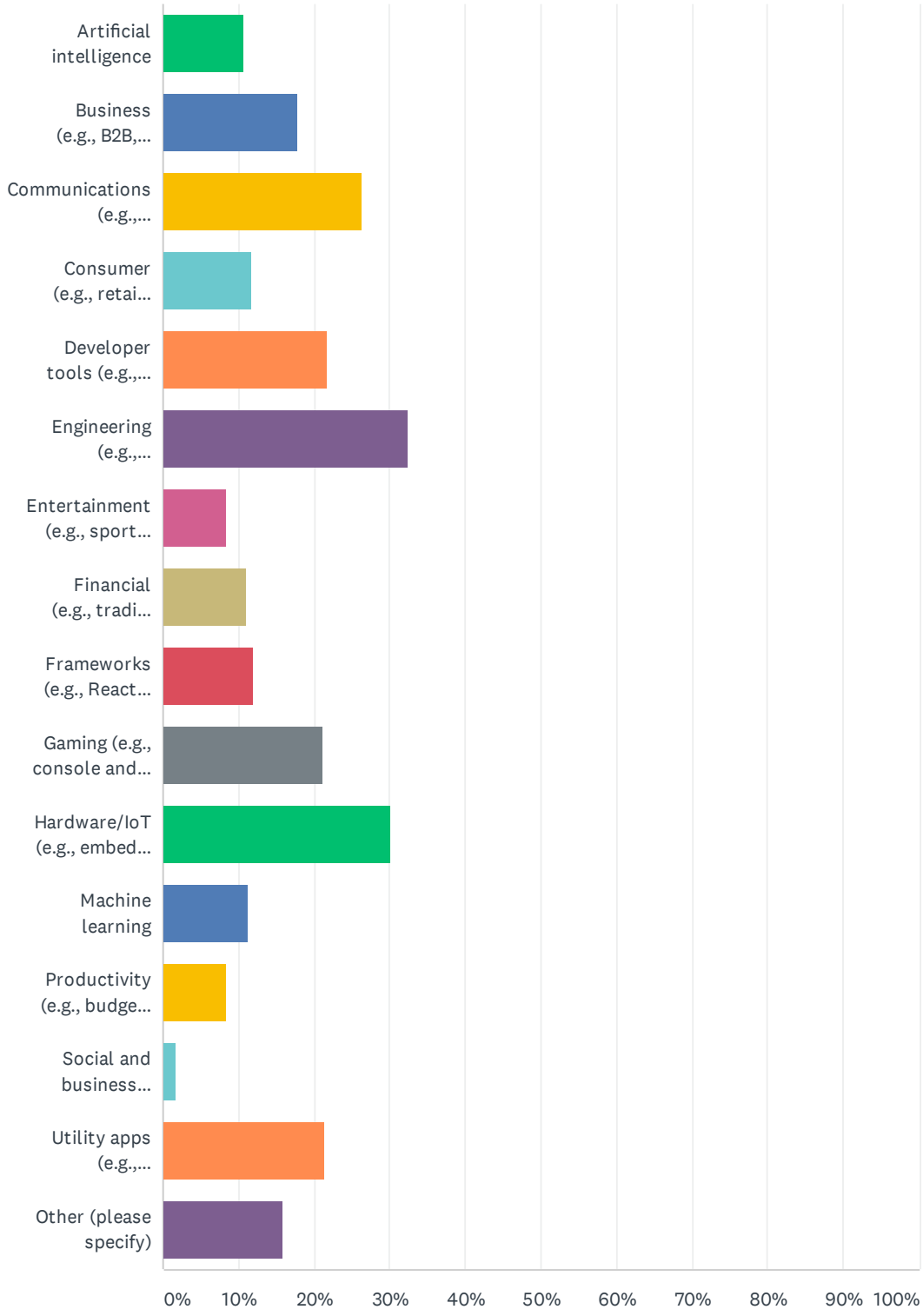
Answered: 1,027 Skipped: 8



ANSWER CHOICES	RESPONSES	
1-2 years	2.04%	21
3-5 years	11.39%	117
6-10 years	20.55%	211
10-20 years	29.11%	299
>20 years	36.90%	379
TOTAL		1,027

Q4 What types of projects do you work on? (select all that apply)

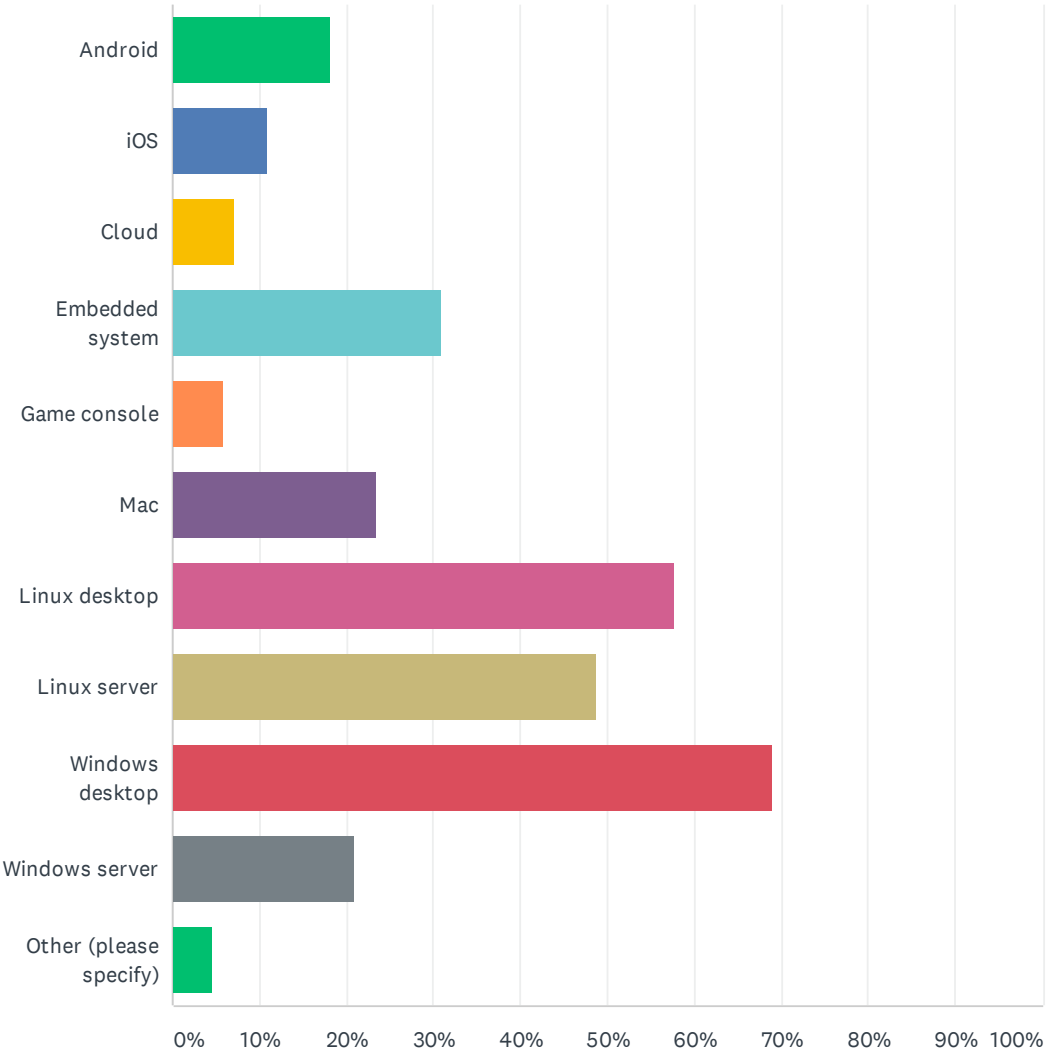
Answered: 1,029 Skipped: 6



ANSWER CHOICES	RESPONSES	
Artificial intelligence	10.59%	109
Business (e.g., B2B, B2E)	17.88%	184
Communications (e.g., networking, email)	26.34%	271
Consumer (e.g., retail websites, mobile apps)	11.66%	120
Developer tools (e.g., compilers, code editors)	21.77%	224
Engineering (e.g., avionics, power management)	32.36%	333
Entertainment (e.g., sports apps, video streaming)	8.45%	87
Financial (e.g., trading, mortgage, asset management)	11.18%	115
Frameworks (e.g., React, Unity)	11.86%	122
Gaming (e.g., console and mobile games)	21.19%	218
Hardware/IoT (e.g., embedded systems, home automation)	30.22%	311
Machine learning	11.37%	117
Productivity (e.g., budget tracking, note taking)	8.45%	87
Social and business networking (e.g., Facebook, Twitter)	1.65%	17
Utility apps (e.g., calculators, simple image editors)	21.38%	220
Other (please specify)	15.84%	163
Total Respondents: 1,029		

Q5 What platforms do you develop for? (select all that apply)

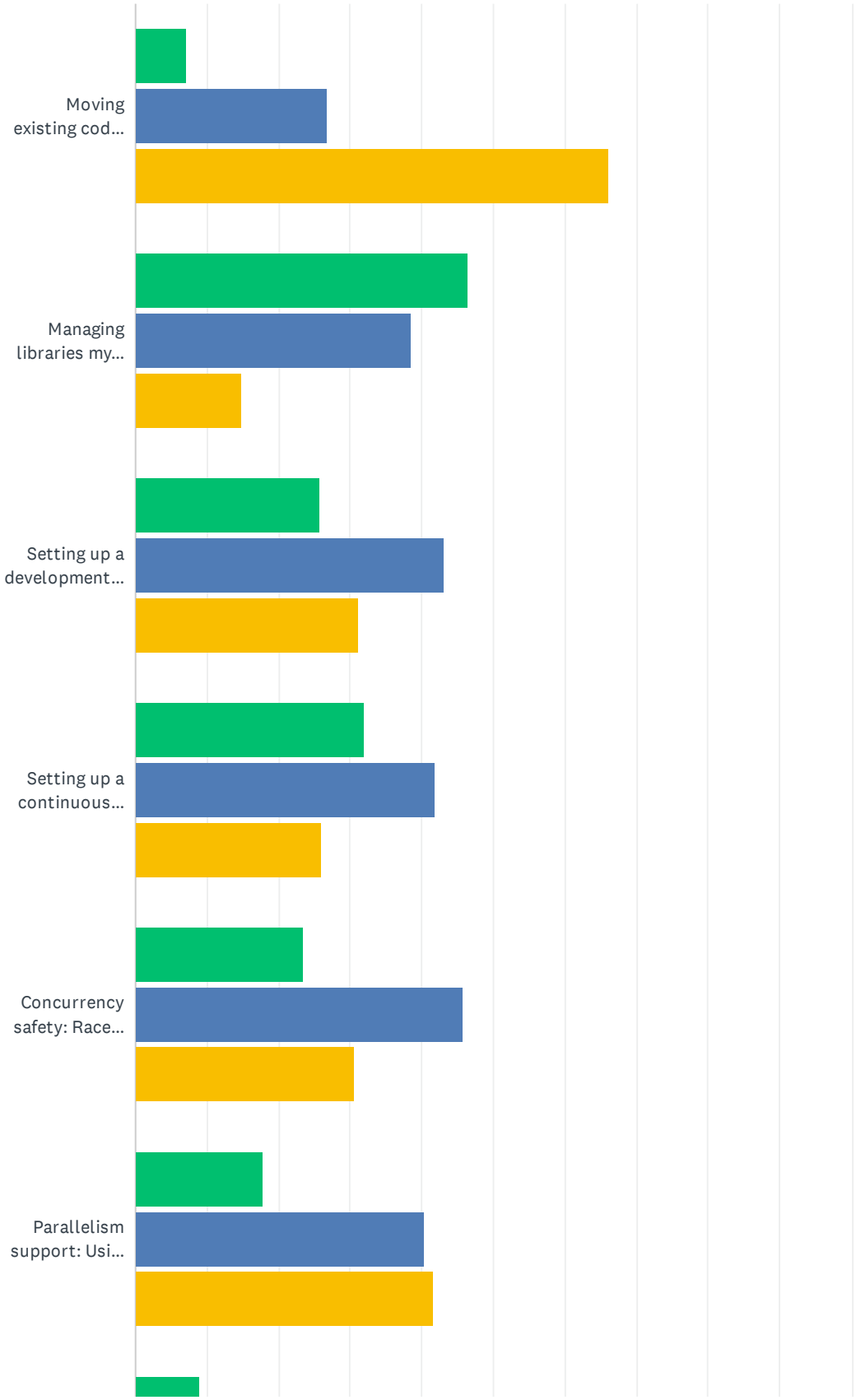
Answered: 1,031 Skipped: 4

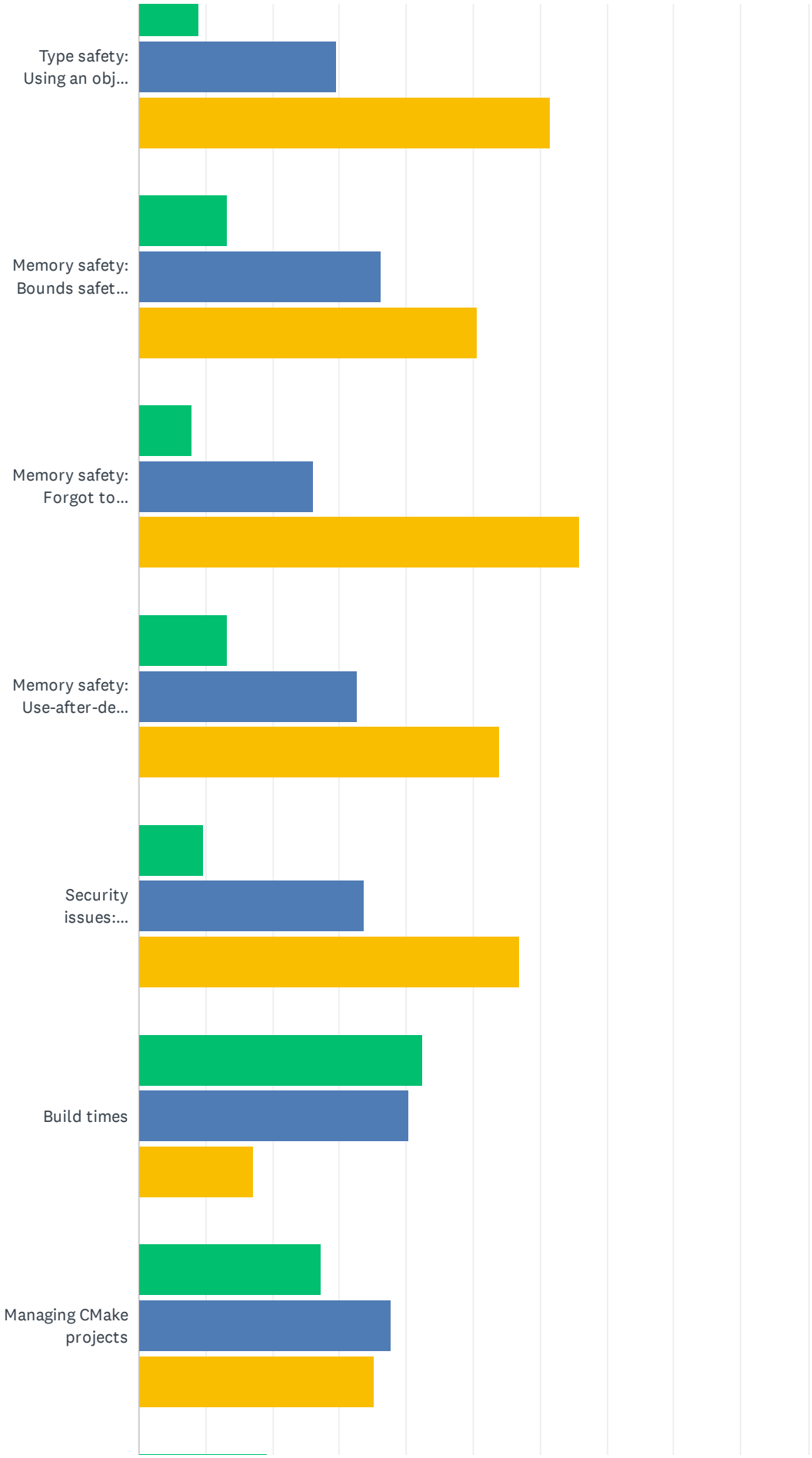


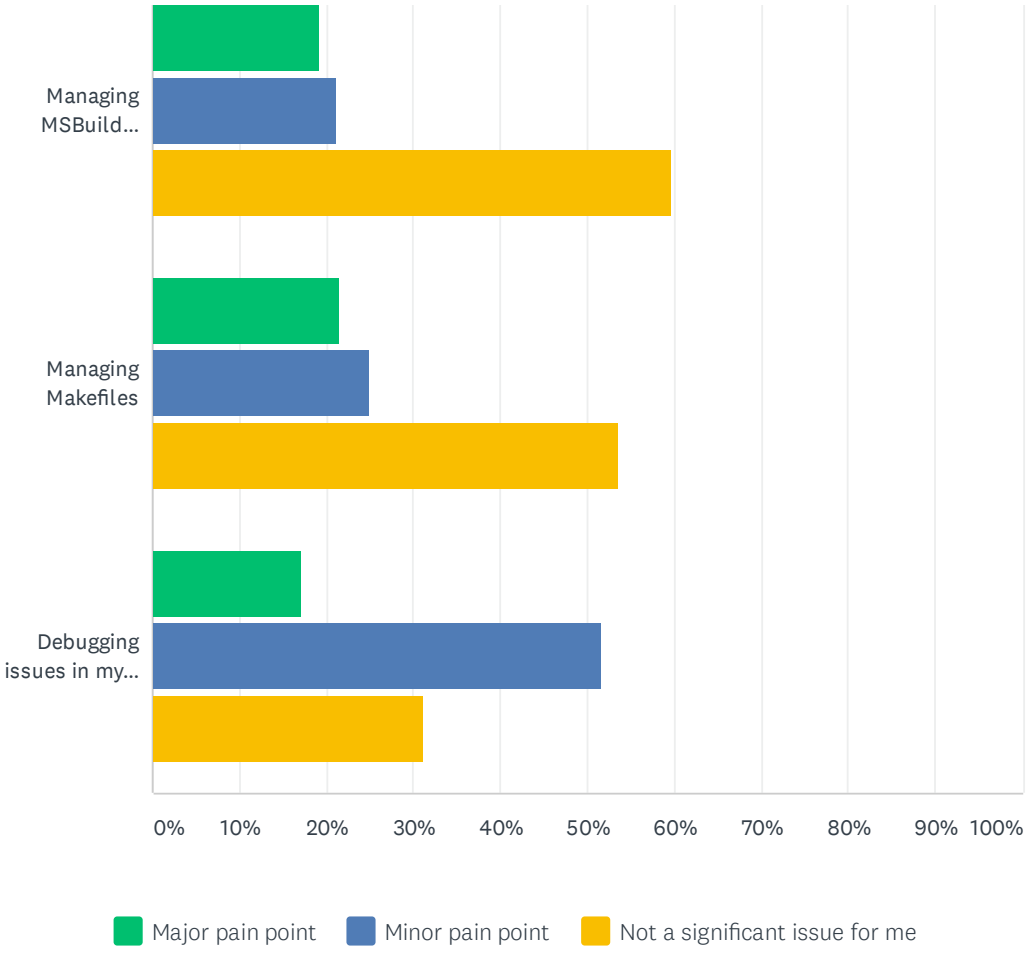
ANSWER CHOICES	RESPONSES	
Android	18.23%	188
iOS	10.86%	112
Cloud	7.18%	74
Embedded system	30.94%	319
Game console	5.82%	60
Mac	23.47%	242
Linux desktop	57.71%	595
Linux server	48.69%	502
Windows desktop	68.96%	711
Windows server	20.85%	215
Other (please specify)	4.56%	47
Total Respondents: 1,031		

Q6 Which of these do you find frustrating about C++ development?

Answered: 1,032 Skipped: 3



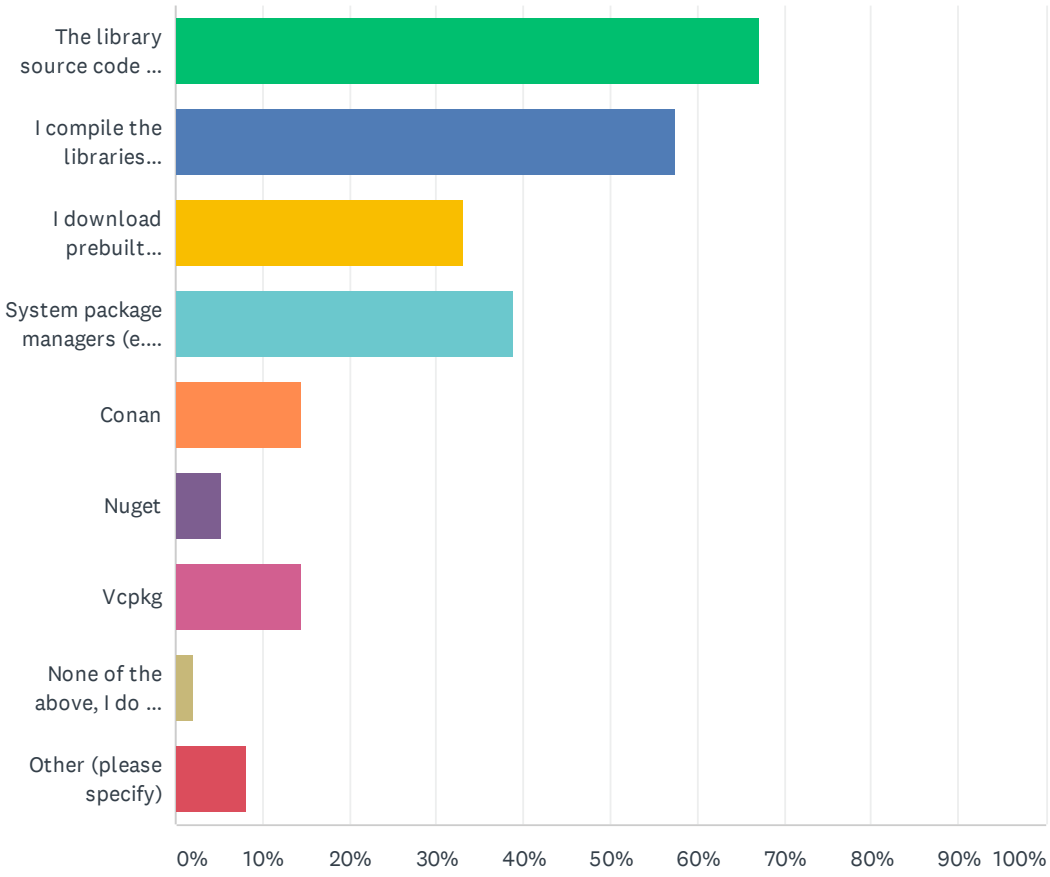

Type safety: Using an obj...
Memory safety: Bounds safet...
Memory safety: Forgot to...
Memory safety: Use-after-de...
Security issues:...
Build times
Managing CMake projects


Managing MSBuild...
Managing Makefiles
Debugging issues in my...
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%
Major pain point
Minor pain point
Not a significant issue for me

	MAJOR PAIN POINT	MINOR PAIN POINT	NOT A SIGNIFICANT ISSUE FOR ME	TOTAL	WEIGHTED AVERAGE
Moving existing code to the latest language standard	7.03% 72	26.76% 274	66.21% 678	1,024	3.25
Managing libraries my application depends on	46.54% 478	38.56% 396	14.90% 153	1,027	1.83
Setting up a development environment from scratch (compiler, build system, IDE, …)	25.71% 263	43.11% 441	31.18% 319	1,023	2.37
Setting up a continuous integration pipeline from scratch (automated builds, tests, …)	32.09% 326	41.93% 426	25.98% 264	1,016	2.20
Concurrency safety: Races, deadlocks, performance bottlenecks	23.51% 241	45.85% 470	30.63% 314	1,025	2.38
Parallelism support: Using more CPU/GPU/other cores to compute an answer faster	17.81% 181	40.45% 411	41.73% 424	1,016	2.66
Type safety: Using an object as the wrong type (unsafe downcasts, unsafe unions, …)	9.08% 93	29.49% 302	61.43% 629	1,024	3.14
Memory safety: Bounds safety issues (read/write beyond the bounds of an object or array)	13.27% 136	36.20% 371	50.54% 518	1,025	2.88
Memory safety: Forgot to delete/free (memory leaks)	7.94% 81	26.18% 267	65.88% 672	1,020	3.24
Memory safety: Use-after-delete/free (dangling pointers, iterators, spans, …)	13.25% 135	32.68% 333	54.07% 551	1,019	2.95
Security issues: Overlaps with “safety” but includes other issues (secret disclosure, vulnerabilities, exploits, …)	9.57% 97	33.63% 341	56.80% 576	1,014	3.04
Build times	42.47% 434	40.41% 413	17.12% 175	1,022	1.92
Managing CMake projects	27.23% 272	37.64% 376	35.14% 351	999	2.43
Managing MSBuild projects	19.20% 187	21.15% 206	59.65% 581	974	3.00
Managing Makefiles	21.57% 212	24.92% 245	53.51% 526	983	2.85
Debugging issues in my code	17.23% 174	51.68% 522	31.09% 314	1,010	2.45

Q7 How do you manage your C++ 1st and 3rd party libraries? (Check all that apply)

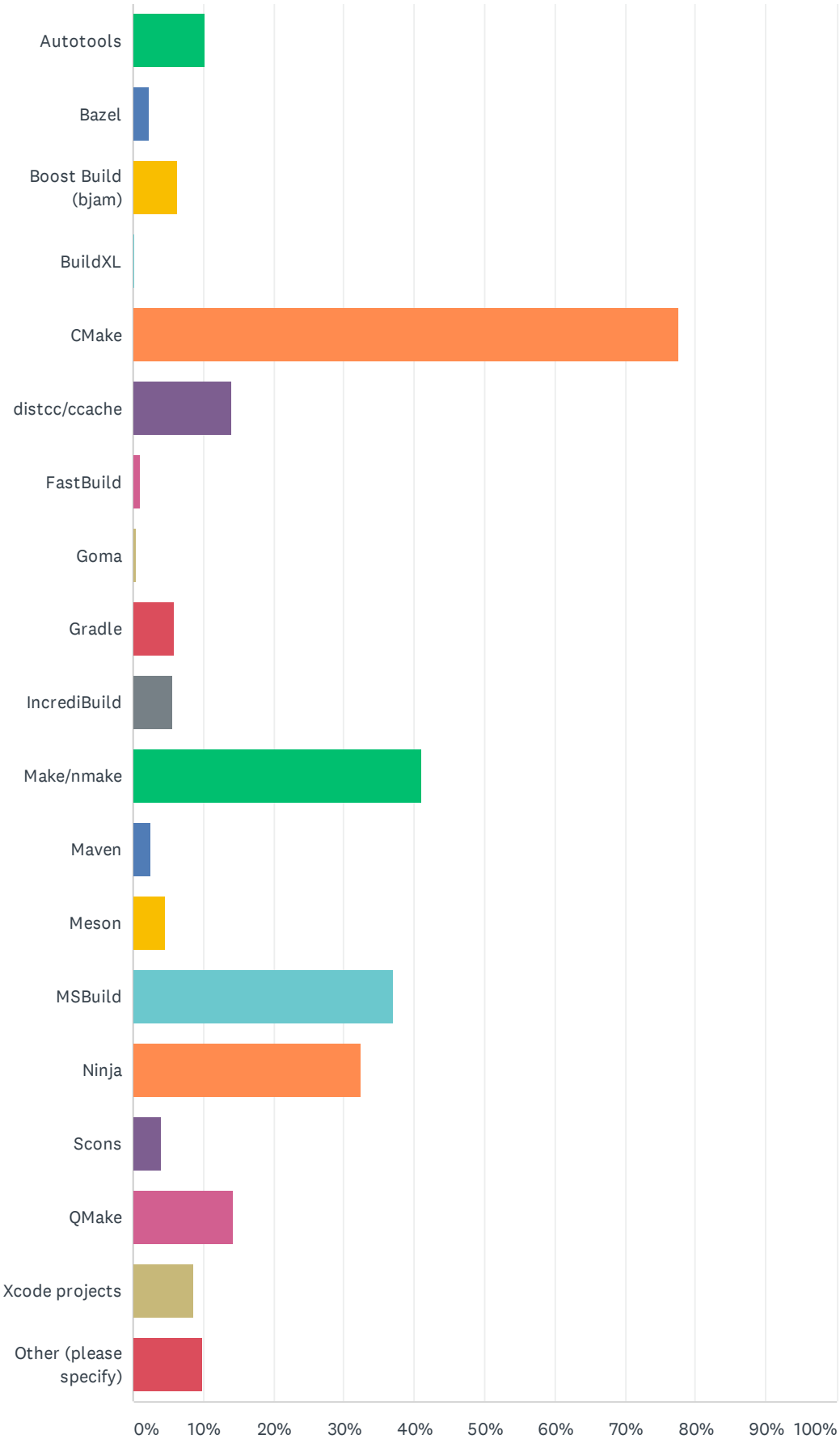
Answered: 1,029 Skipped: 6



ANSWER CHOICES	RESPONSES	
The library source code is part of my build	67.06%	690
I compile the libraries separately using their instructions	57.63%	593
I download prebuilt libraries from the Internet	33.04%	340
System package managers (e.g., apt, brew, …)	38.87%	400
Conan	14.48%	149
Nuget	5.25%	54
Vcpkg	14.38%	148
None of the above, I do not have any dependencies	2.04%	21
Other (please specify)	8.16%	84
Total Respondents: 1,029		

Q8 What build tools do you use? (Check all that apply)

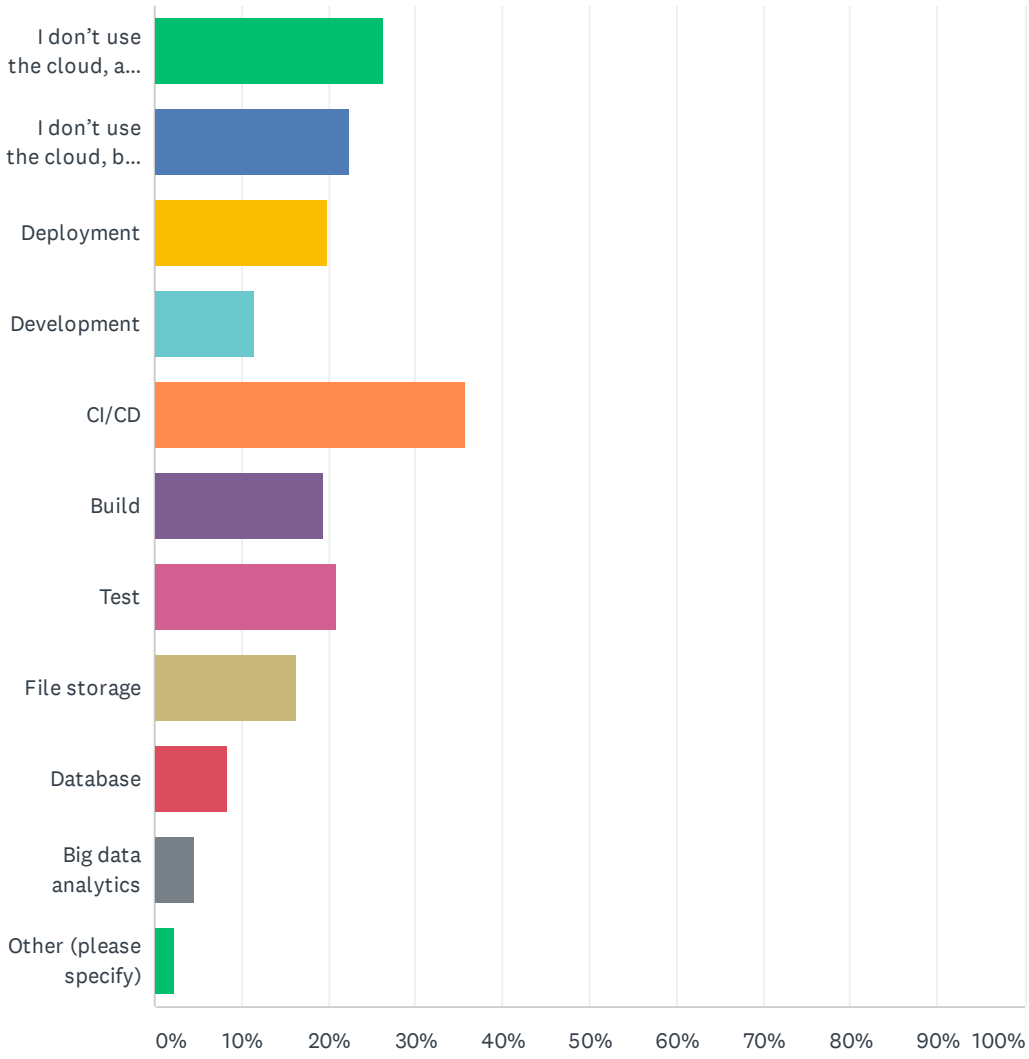
Answered: 1,015 Skipped: 20


Autotools
Bazel
Boost Build (bjam)
BuildXL
CMake
distcc/ccache
FastBuild
Goma
Gradle
IncrediBuild
Make/nmake
Maven
Meson
MSBuild
Ninja
Scons
QMake
Xcode projects
Other (please specify)
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%

ANSWER CHOICES	RESPONSES	
Autotools	10.25%	104
Bazel	2.27%	23
Boost Build (bjam)	6.21%	63
BuildXL	0.20%	2
CMake	77.64%	788
distcc/ccache	13.99%	142
FastBuild	1.08%	11
Goma	0.39%	4
Gradle	5.81%	59
IncrediBuild	5.62%	57
Make/nmake	40.99%	416
Maven	2.46%	25
Meson	4.53%	46
MSBuild	37.04%	376
Ninja	32.41%	329
Scons	4.04%	41
QMake	14.19%	144
Xcode projects	8.67%	88
Other (please specify)	9.75%	99
Total Respondents: 1,015		

Q9 What parts of your development lifecycle use the cloud? (Check all that apply)

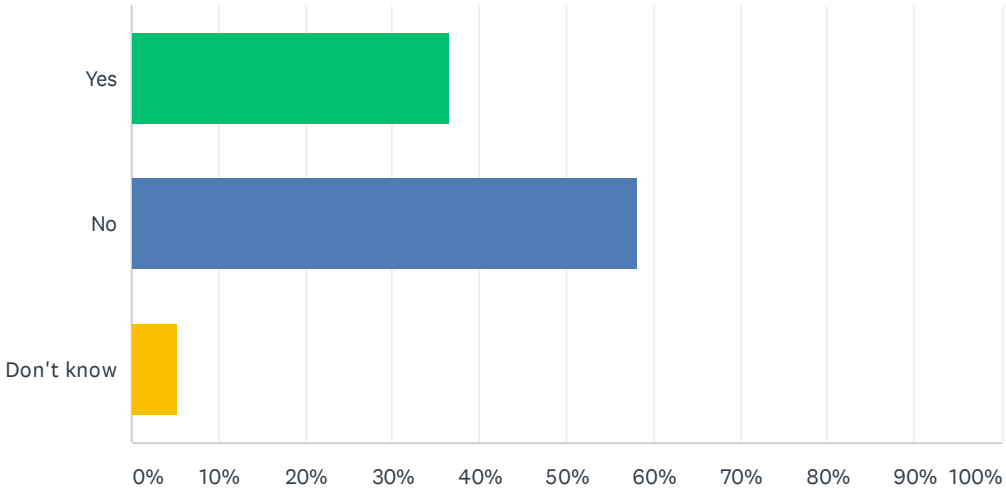
Answered: 1,017 Skipped: 18



ANSWER CHOICES	RESPONSES	
I don’t use the cloud, and I am not interested	26.35%	268
I don’t use the cloud, but I am interested	22.42%	228
Deployment	19.86%	202
Development	11.60%	118
CI/CD	35.79%	364
Build	19.47%	198
Test	20.94%	213
File storage	16.32%	166
Database	8.46%	86
Big data analytics	4.52%	46
Other (please specify)	2.26%	23
Total Respondents: 1,017		

Q10 Does your current project use sanitizers and/or fuzzing as part of its normal development and release cycle?

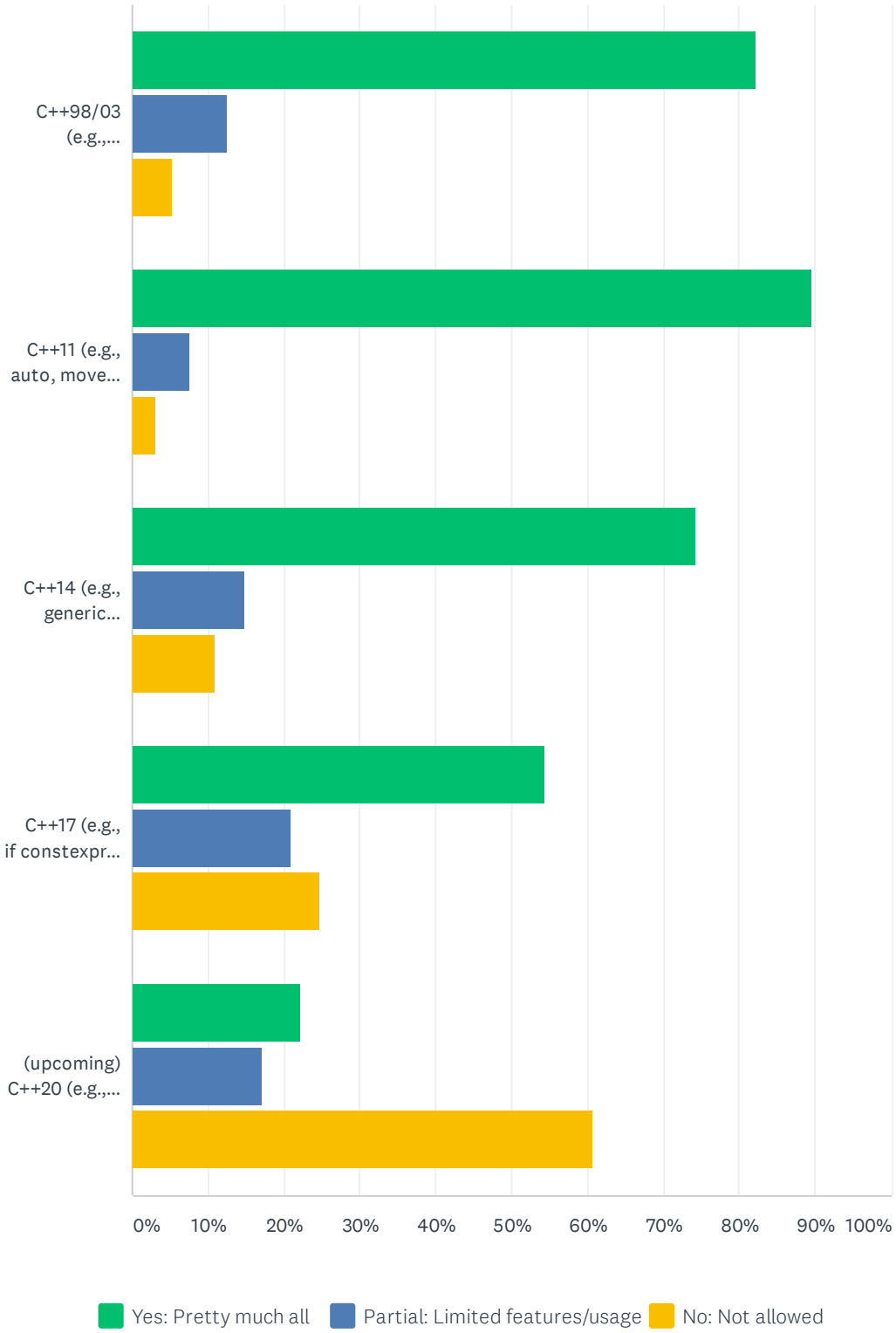
Answered: 1,024 Skipped: 11



ANSWER CHOICES	RESPONSES	
Yes	36.62%	375
No	58.11%	595
Don't know	5.27%	54
TOTAL		1,024

Q11 What version(s) of C++ are you allowed to use on your current project (work or school)?

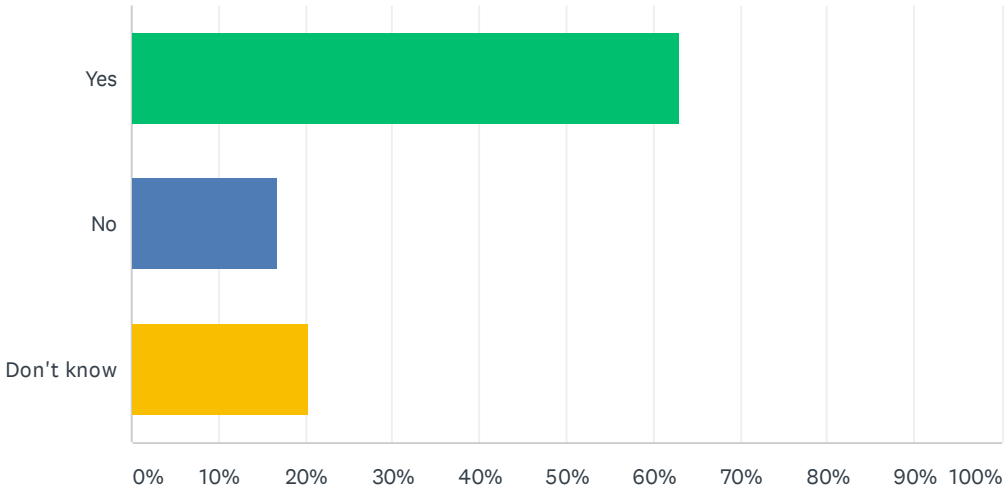
Answered: 1,025 Skipped: 10



	YES: PRETTY MUCH ALL	PARTIAL: LIMITED FEATURES/USAGE	NO: NOT ALLOWED	TOTAL	WEIGHTED AVERAGE
C++98/03 (e.g., exceptions, templates, RTTI)	82.27% 784	12.59% 120	5.14% 49	953	2.77
C++11 (e.g., auto, move semantics, =delete/=default, shared_ptr, lambdas)	89.45% 873	7.48% 73	3.07% 30	976	2.86
C++14 (e.g., generic lambdas, auto return types, general constexpr functions)	74.31% 726	14.84% 145	10.85% 106	977	2.63
C++17 (e.g., if constexpr, if/switch scoped variables, structured bindings, string_view, optional/any/variant, Parallel STL)	54.45% 545	20.88% 209	24.68% 247	1,001	2.30
(upcoming) C++20 (e.g., concepts, coroutines, modules)	22.12% 213	17.24% 166	60.64% 584	963	1.61

Q12 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)?

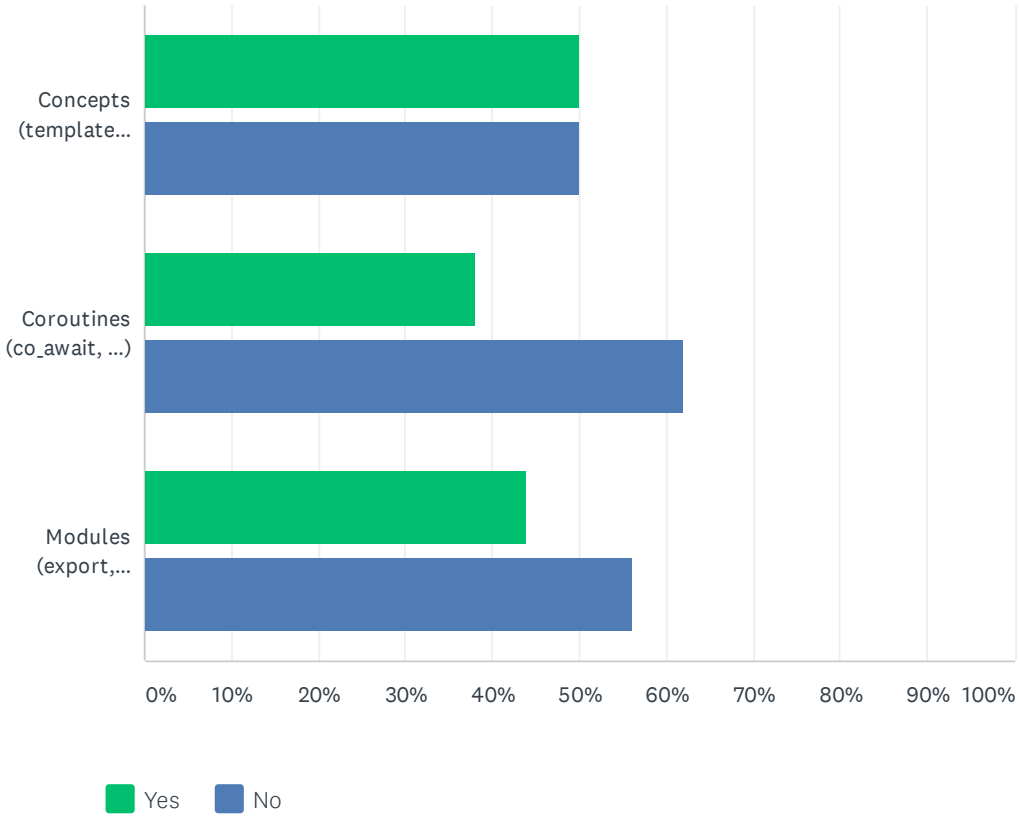
Answered: 1,022 Skipped: 13



ANSWER CHOICES	RESPONSES	
Yes	62.92%	643
No	16.83%	172
Don't know	20.25%	207
TOTAL		1,022

Q13 Specifically for upcoming major C++20 features: In the next 12 months, does your current project plan to allow use of these draft C++20 features?

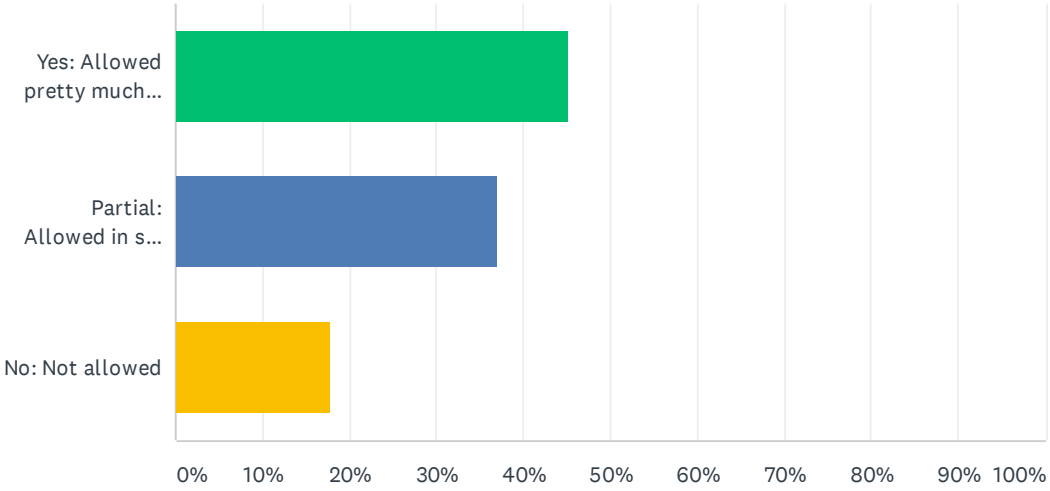
Answered: 1,006 Skipped: 29



	YES	NO	TOTAL	WEIGHTED AVERAGE
Concepts (template constraints, requires, ...)	49.90% 499	50.10% 501	1,000	2.00
Coroutines (co_await, ...)	38.10% 378	61.90% 614	992	2.24
Modules (export, import, ...)	43.95% 436	56.05% 556	992	2.12

Q14 Is throwing exceptions to report errors allowed throughout your current project?

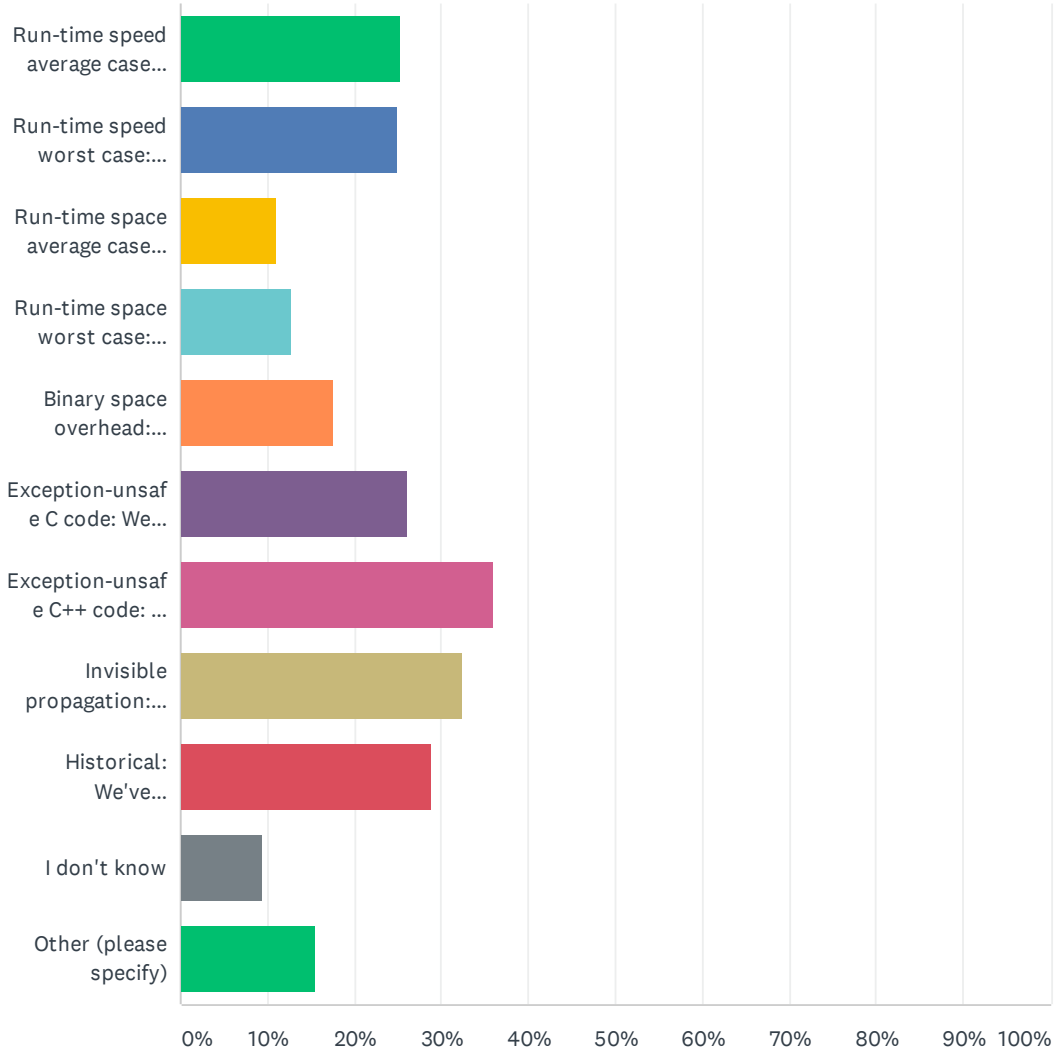
Answered: 1,028 Skipped: 7



ANSWER CHOICES	RESPONSES	
Yes: Allowed pretty much everywhere	45.14%	464
Partial: Allowed in some parts of the code but not others	37.06%	381
No: Not allowed	17.80%	183
TOTAL		1,028

Q15 If you answered "Partial" or "No" on the previous question: When throwing exceptions not allowed, what are the reasons? (select all that apply)

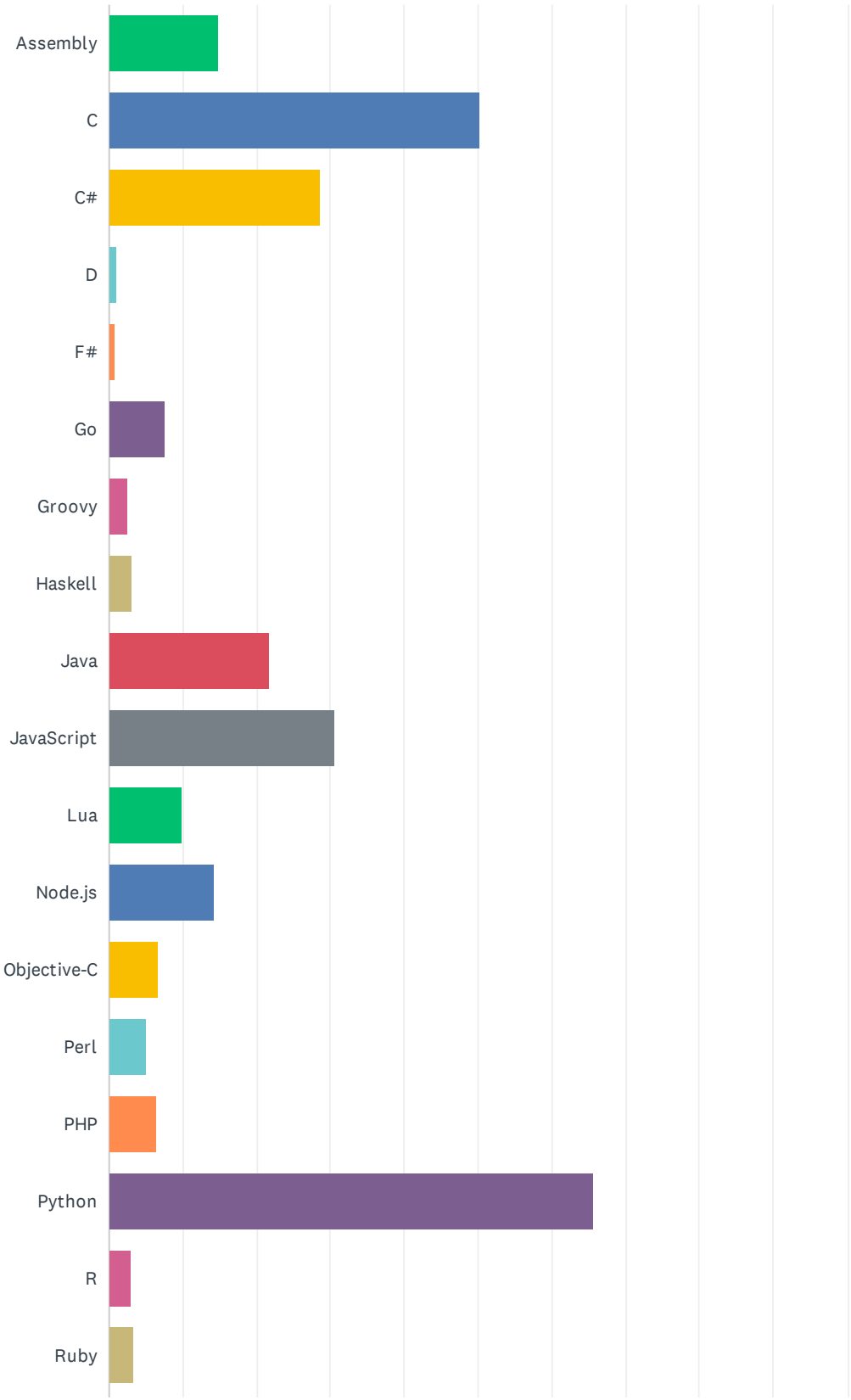
Answered: 572 Skipped: 463

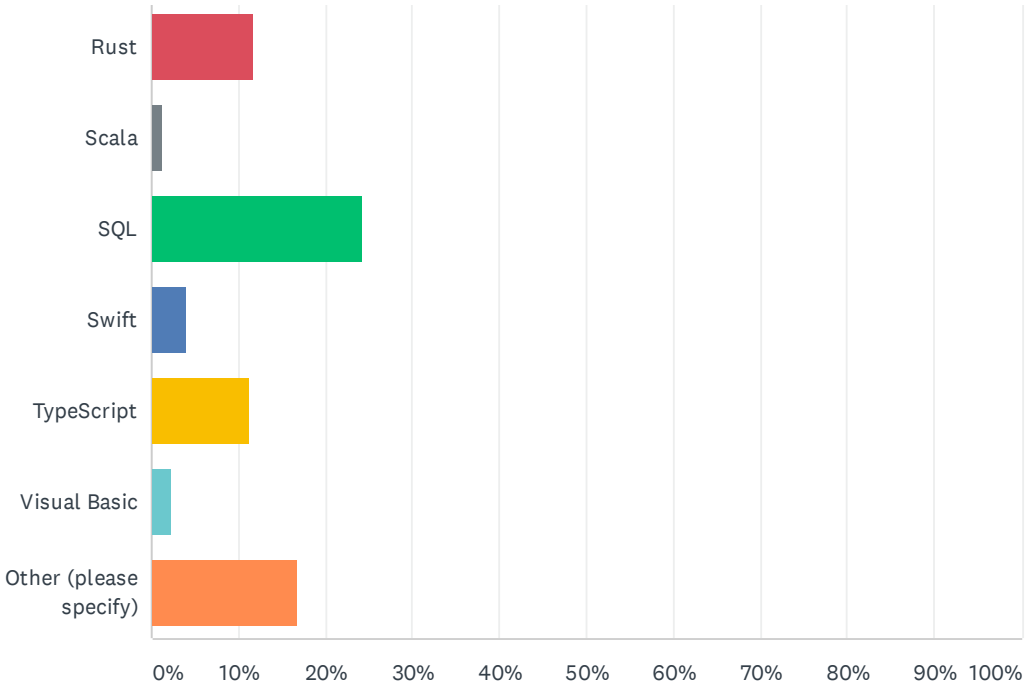


ANSWER CHOICES	RESPONSES	
Run-time speed average case: Throwing is just slow most of the time, it would need to be much faster than now nearly all of the time (it would be okay if very rarely throwing might take a long time)	25.35%	145
Run-time speed worst case: Throwing speed is too unpredictable, some of our code needs a guaranteed upper limit on how long reporting an error will take	25.00%	143
Run-time space average case: Throwing takes too much stack/heap space in general	11.19%	64
Run-time space worst case: Throwing uses unpredictable stack/heap space, some of our code needs a guaranteed upper limit on how much space reporting an error will use, or a guarantee that throwing won't heap-allocate at all	12.76%	73
Binary space overhead: Enabling exceptions makes my binaries too much bigger	17.66%	101
Exception-unsafe C code: We can't throw through C code because it wasn't written with exceptions in mind and therefore is likely to break	26.05%	149
Exception-unsafe C++ code: We can't throw through some C++ code we think possibly wasn't written in an exception-correct way and therefore is likely to break	36.01%	206
Invisible propagation: The exceptional control paths are invisible in source code, so they're too hard to make reliable	32.52%	186
Historical: We've restricted throwing for a long time and haven't revisited that decision lately	28.85%	165
I don't know	9.44%	54
Other (please specify)	15.56%	89
Total Respondents: 572		

Q16 Besides C++, what programming languages/environments do you use in your current and recent projects? (select all that apply)

Answered: 1,003 Skipped: 32

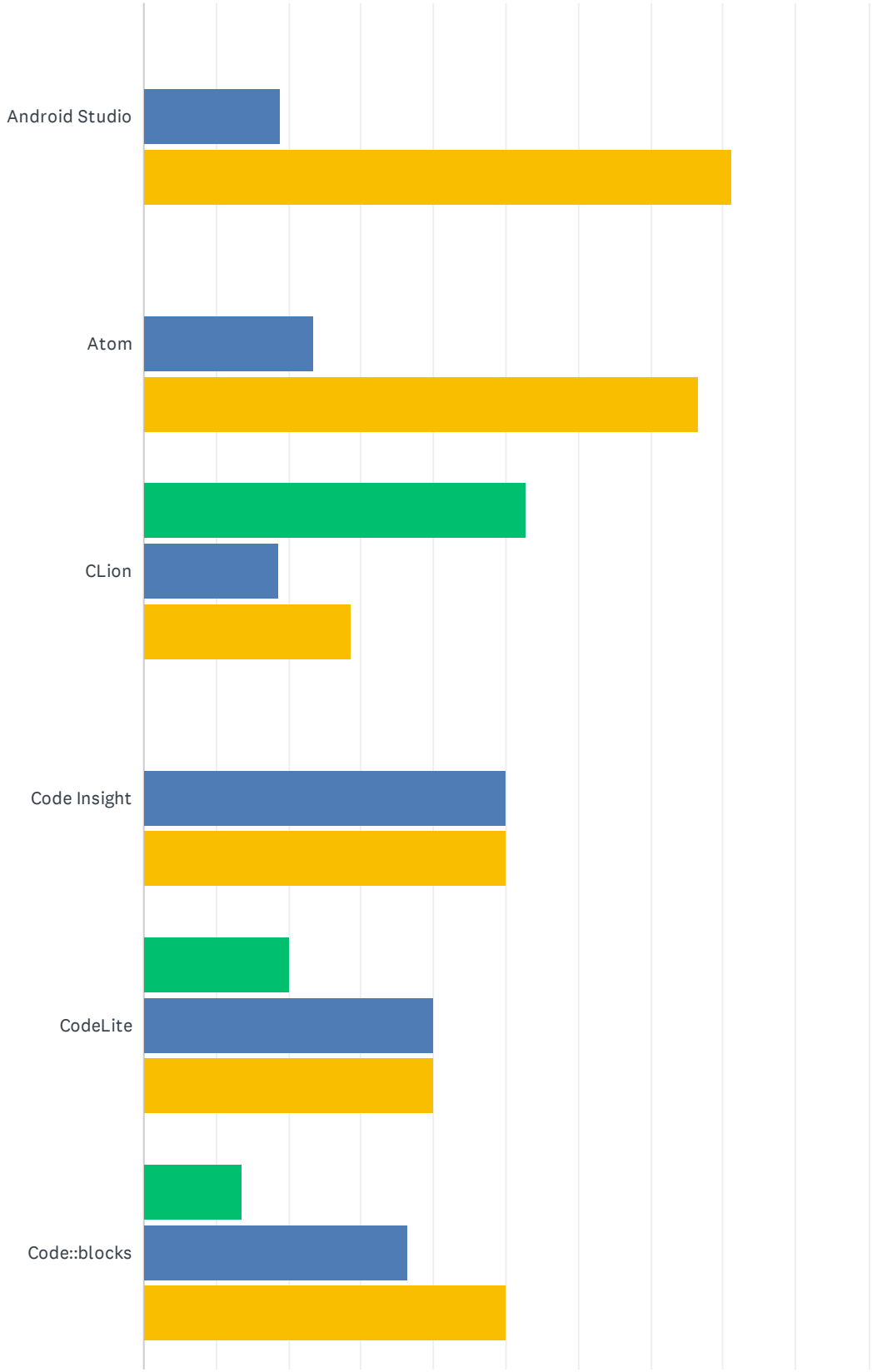


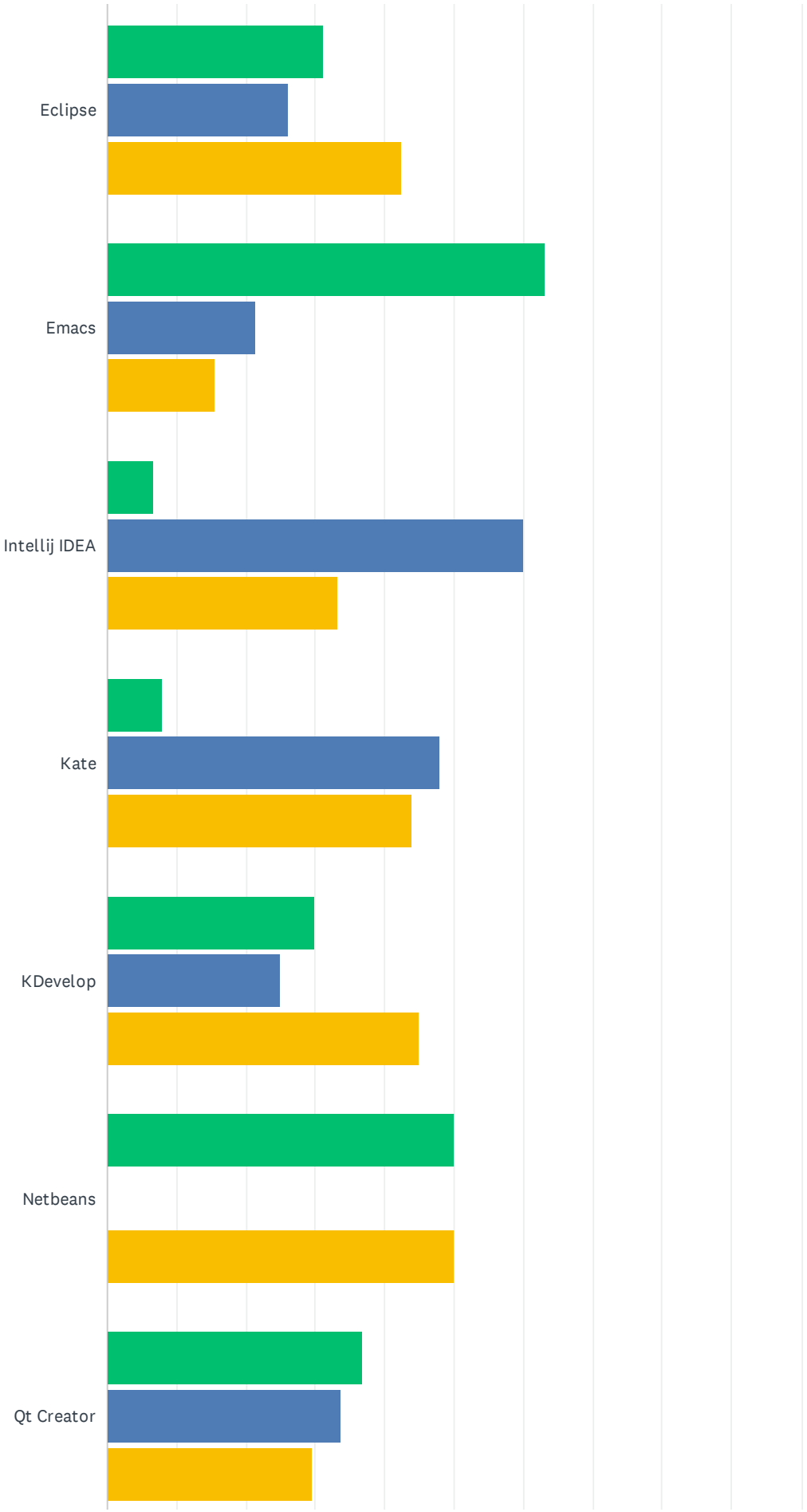

Rust
Scala
SQL
Swift
TypeScript
Visual Basic
Other (please specify)
0%
10%
20%
30%
40%
50%
60%
70%
80%
90%
100%

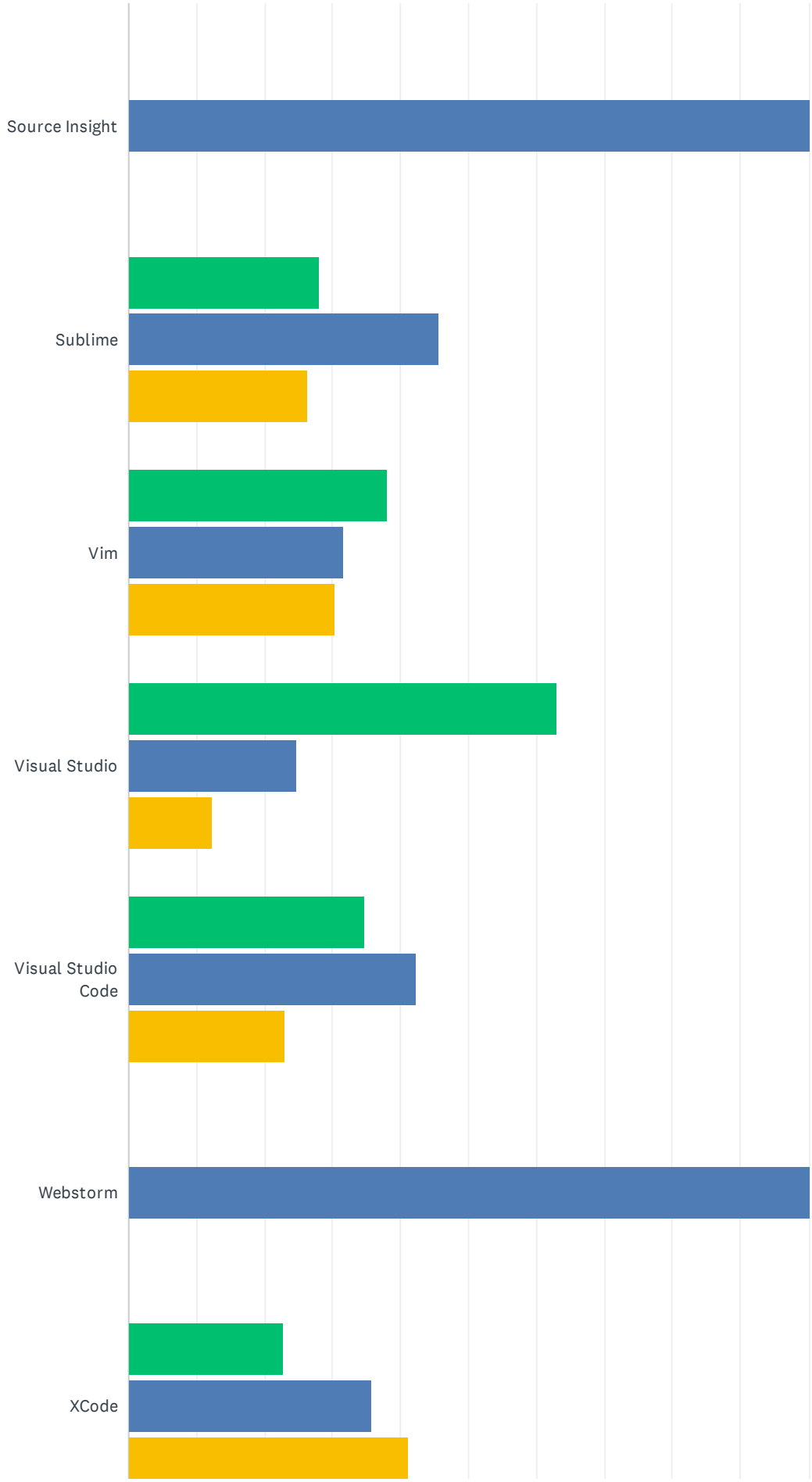
ANSWER CHOICES	RESPONSES	
Assembly	14.86%	149
C	50.25%	504
C#	28.61%	287
D	1.00%	10
F#	0.90%	9
Go	7.58%	76
Groovy	2.49%	25
Haskell	3.09%	31
Java	21.73%	218
JavaScript	30.51%	306
Lua	9.77%	98
Node.js	14.16%	142
Objective-C	6.68%	67
Perl	4.99%	50
PHP	6.38%	64
Python	65.70%	659
R	2.89%	29
Ruby	3.29%	33
Rust	11.76%	118
Scala	1.30%	13
SQL	24.33%	244
Swift	3.99%	40
TypeScript	11.27%	113
Visual Basic	2.39%	24
Other (please specify)	16.65%	167
Total Respondents: 1,003		

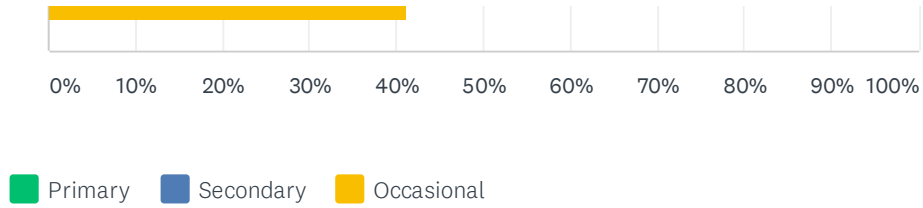
Q17 Which development environments (IDEs) or editors do you use for C++ development?

Answered: 1,009 Skipped: 26




Eclipse
Emacs
Intellij IDEA
Kate
KDevelop
Netbeans
Qt Creator


Source Insight
Sublime
Vim
Visual Studio
Visual Studio Code
Webstorm
XCode



	PRIMARY	SECONDARY	OCCASIONAL	TOTAL	WEIGHTED AVERAGE
Android Studio	0.00% 0	18.75% 9	81.25% 39	48	2.81
Atom	0.00% 0	23.53% 4	76.47% 13	17	2.76
CLion	52.66% 99	18.62% 35	28.72% 54	188	1.76
Code Insight	0.00% 0	50.00% 1	50.00% 1	2	2.50
CodeLite	20.00% 1	40.00% 2	40.00% 2	5	2.20
Code::blocks	13.64% 3	36.36% 8	50.00% 11	22	2.36
Eclipse	31.25% 25	26.25% 21	42.50% 34	80	2.11
Emacs	63.11% 65	21.36% 22	15.53% 16	103	1.52
Intellij IDEA	6.67% 1	60.00% 9	33.33% 5	15	2.27
Kate	8.00% 2	48.00% 12	44.00% 11	25	2.36
KDevelop	30.00% 6	25.00% 5	45.00% 9	20	2.15
Netbeans	50.00% 6	0.00% 0	50.00% 6	12	2.00
Qt Creator	36.77% 82	33.63% 75	29.60% 66	223	1.93
Source Insight	0.00% 0	100.00% 2	0.00% 0	2	2.00
Sublime	28.07% 16	45.61% 26	26.32% 15	57	1.98
Vim	38.03% 135	31.55% 112	30.42% 108	355	1.92
Visual Studio	63.02% 351	24.60% 137	12.39% 69	557	1.49
Visual Studio Code	34.76% 162	42.27% 197	22.96% 107	466	1.88
Webstorm	0.00% 0	100.00% 3	0.00% 0	3	2.00
XCode	22.83% 21	35.87% 33	41.30% 38	92	2.18

Q18 If you could wave a magic wand and change one thing about any part of C++ or C++ standardization, what would it be, and how would that change help your daily work?

Answered: 632 Skipped: 403