Test-Case Reduction for C Compiler Bugs

J. Regehr et al., PLDI ’12

Presenter: Hyunsu Kim

Compilers do have bugs

Bugzilla – Bug List

Home | New | Browse | Search | [?] | Reports | Help | Log In | Forgot Password

New user self-registration is disabled due to spam. For an account please email bugs-admin@lists.llvm.org with your e-mail address and full name.

Tue Oct 6 2020 13:26:40 PDT

We guarantee you'll be satisfied, or will give you twice your money back!

Hide Search Description

Component: C **Resolution:** --- **Product:** clang

88 bugs found.

ID	Product	Comp	Assignee	Status	Resolution	Summary	Changed
46792	clang	C	efriedma	NEW	---	ARM long double NaN cannot be negated	2020-07-21
45157	clang	C	ndesaulniers	NEW	---	Clang and gcc disagree on whether a `const struct` is a compile-time constant	2020-03-12
42034	clang	C	unassignedclangbugs	NEW	---	Incorrect expansion of '@' in inline asm macros	2019-05-27
42098	clang	C	unassignedclangbugs	NEW	---	__CHAR16_TYPE__ and __CHAR32_TYPE__ are defined on MacOS but they don't actually work	2019-06-01
42120	clang	C	unassignedclangbugs	NEW	---	Inconsistent (missing) incompatible function pointer type warnings	2019-06-04
42142	clang	C	unassignedclangbugs	NEW	---	Missing switch case checking in Clang diagnostics	2019-11-12
42554	clang	C	unassignedclangbugs	NEW	---	__BIGGEST_ALIGNMENT__ not the same as gcc for AVX and later	2019-07-09
42610	clang	C	unassignedclangbugs	NEW	---	No warning on use of uninitialized struct member	2020-02-12
42625	clang	C	unassignedclangbugs	NEW	---	Uninitialized warning missing for struct member access inside GNU C statement expression	2019-09-04
42709	clang	C	unassignedclangbugs	NEW	---	Extra stack load/store generated for a volatile {i16, i16} store	2019-07-22
42996	clang	C	unassignedclangbugs	NEW	---	Inline assembly incompatibility with gcc for array parameters	2019-08-14
43027	clang	C	unassignedclangbugs	NEW	---	clang accepts 'enum struct' in C mode with warning despite being unusable	2019-12-11
43089	clang	C	unassignedclangbugs	NEW	---	clang's inline assembler translates mov %xmm0, (%rdi) into an movq	2019-08-22
43104	clang	C	unassignedclangbugs	NEW	---	missed opt: restrict not propagated to definition	2020-01-27
43114	clang	C	unassignedclangbugs	NEW	---	Clan7.0.1 does not apply -D_FORTIFY_SOURCE=2	2019-08-26
43117	clang	C	unassignedclangbugs	NEW	---	clang crashes on x86_64-linux-gnu on invalid code of a flexible array member in a union	2019-08-26
43262	clang	C	unassignedclangbugs	NEW	---	clang crash on specific source code.	2019-09-10
43331	clang	C	unassignedclangbugs	NEW	---	[ARM64:NEON] parentheses required for intrinsics (breaks designated initializers)	2019-09-16
43448	clang	C	unassignedclangbugs	NEW	---	attribute target syntax doesn't match -march options	2019-10-03
43576	clang	C	unassignedclangbugs	NEW	---	FFT and Sparse matmult in Scimark2 are slower with -O2 & -O3 than -O1	2019-10-07
43600	clang	C	unassignedclangbugs	NEW	---	-Wincomplete-setjmp-declaration triggers on typedef	2019-10-07
43603	clang	C	unassignedclangbugs	NEW	---	C11 atomics always yielding library call with -mcpu=cortex-m0plus	2019-10-10
43683	clang	C	unassignedclangbugs	NEW	---	Does clang 9.0.0 version support VS2010?	2019-10-15
43725	clang	C	unassignedclangbugs	NEW	---	Bad floating-point "optimizations"	2020-07-03
43727	clang	C	unassignedclangbugs	NEW	---	Compares involving NaN are done wrong	2019-10-24
43766	clang	C	unassignedclangbugs	NEW	---	In version 8 of clang code gets inf loop at runtime; in version 4 it crashes the compiler	2019-11-11
43777	clang	C	unassignedclangbugs	NEW	---	FP initializer using SNAN in expression	2019-10-23
43969	clang	C	unassignedclangbugs	NEW	---	Clang crashes when incrementing a vector by a float scalar	2019-11-11
43987	clang	C	unassignedclangbugs	NEW	---	Broken module found, compilation aborted	2020-05-19
44101	clang	C	unassignedclangbugs	NEW	---	PowerPC64: clang crash during compilation of crafty	2019-11-22
44224	clang	C	unassignedclangbugs	NEW	---	Assertion `isa<X>(Val) && "cast<Ty>() argument of incompatible type!" failed	2019-12-04
44360	clang	C	unassignedclangbugs	NEW	---	int have_read = read (...) --- unable to deduce that "have_read >= -1"	2019-12-21
44391	clang	C	unassignedclangbugs	NEW	---	[Aarch64] Wshadow and vcopy intrinsic	2020-01-01
44406	clang	C	unassignedclangbugs	NEW	---	file-scoped variably modified type erroneously supported in C as a GNU extension (GCC provides the extension only for C++)	2020-01-06
44480	clang	C	unassignedclangbugs	NEW	---	__FILE__ is modified whenever "#include __FILE__" is encountered	2020-01-07
44524	clang	C	unassignedclangbugs	NEW	---	Pruning of dead -Wshift-count-overflow suppressed in dead branches of _Generic	2020-01-12
44535	clang	C	unassignedclangbugs	NEW	---	ASAN and Visual Studio 2019 missing translation unit name	2020-01-13
44572	clang	C	unassignedclangbugs	NEW	---	Should -Werror imply -Wl,--fatal-warnings?	2020-01-16
44607	clang	C	unassignedclangbugs	NEW	---	Relax ARM NEON literal rules	2020-01-21
44855	clang	C	unassignedclangbugs	NEW	---	Combining a pointer to array of const TYPE with an otherwise size-compatible array of TYPE should work and yield only a _pedantic_ warning	2020-02-09

https://bugs.llvm.org/buglist.cgi?component=C&product=clang&resolution=---

Compilers most likely to fail for non-trivial cases

Csmith, a fuzzer for C compiler

Csmith generates..

- 37-279 KB size
- 1.4K LOC
- Not human-friendly

Compilers most likely to fail for non-trivial cases

Csmith, a fuzzer for C compiler

Csmith generates..

- 37-279 KB size
- 1.4K LOC
- Not human-friendly

Compiler developer wants..

- <0.5 KB size
- <100 LOC
- Human-amenable, meaningful

Creduce generates valid test case

Valid test input ⇔ No undefined behavior (UB)

```
1.   int main (void) {
2.     Int x;
3.     x = 2;
4.     return x + 1;
5.   }
```

P_{init}

Creduce generates valid test case

Valid test input ⇔ No undefined behavior (UB)

```
1.   int main (void) {
2.     Int x;
3.     x = 2;
4.     return x + 1;
5.   }
```

P_{init}

Δ

```
1.   int main (void) {
2.     Int x;
3.     return x + 1;
4.   }
```

$P_{reduced}$

Creduce generates valid test case

Valid test input ⇔ No undefined behavior (UB)

```
1.   int main (void) {
2.     Int x;
3.     x = 2;
4.     return x + 1;
5.   }
```

P_{init}

Δ

```
1.   int main (void) {
2.     Int x;
3.     return x + 1;
4.   }
```

$P_{reduced}$

contains UB
(i.e. reading uninitialized)

Creduce generates valid test case

Use semantics-checking C interpreters like

- KCC
- Frama-C

which are

- capable of debugging, catching UB
- involve static analysis

Creduce applies pluggable transformations until fixpoint

Pluggable transformations

- Source-to-source alterations over test case
- E.g. turning `union` into `struct`, shorten var/fun name, copy propagations

Creduce applies pluggable transformations until fixpoint

Pluggable transformations

- Source-to-source alterations over test case
- E.g. turning `union` into `struct`, shorten var/fun name, copy propagations

has three methods

- `new` : `()` → state // **fresh transformation state**
- `transform` : state, testcase → `ok|stop` // **alter testcase**
- `advance` : state, testcase → `()` // **move on to next trial**

Creduce applies pluggable transformations until fixpoint

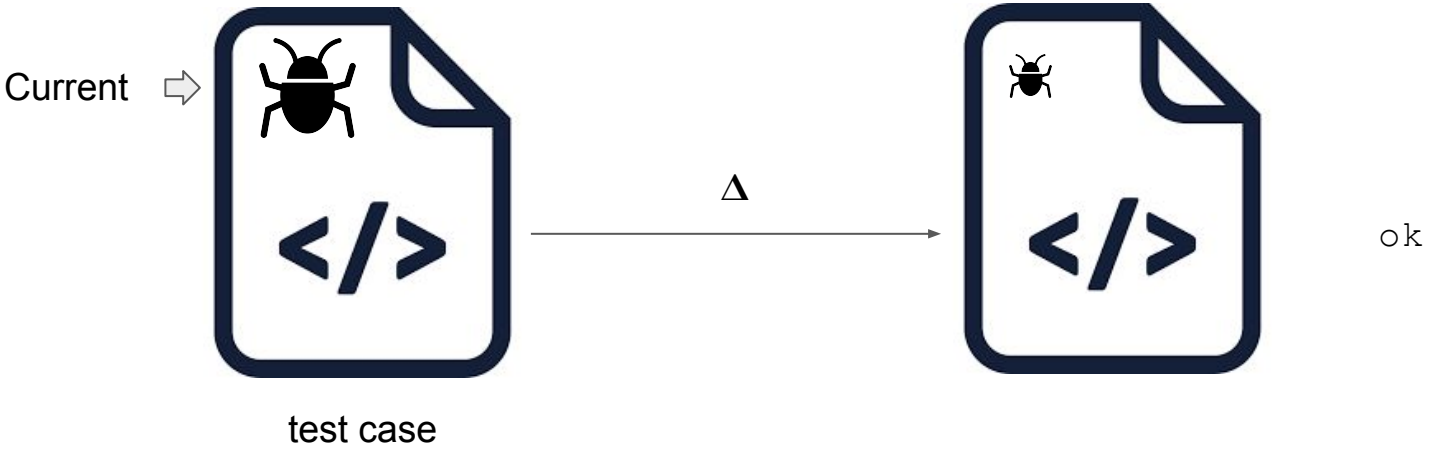
T : transformation

T::new() → (s) state (e.g. cursor indicating loc in testcase)

Creduce applies pluggable transformations until fixpoint

T : transformation

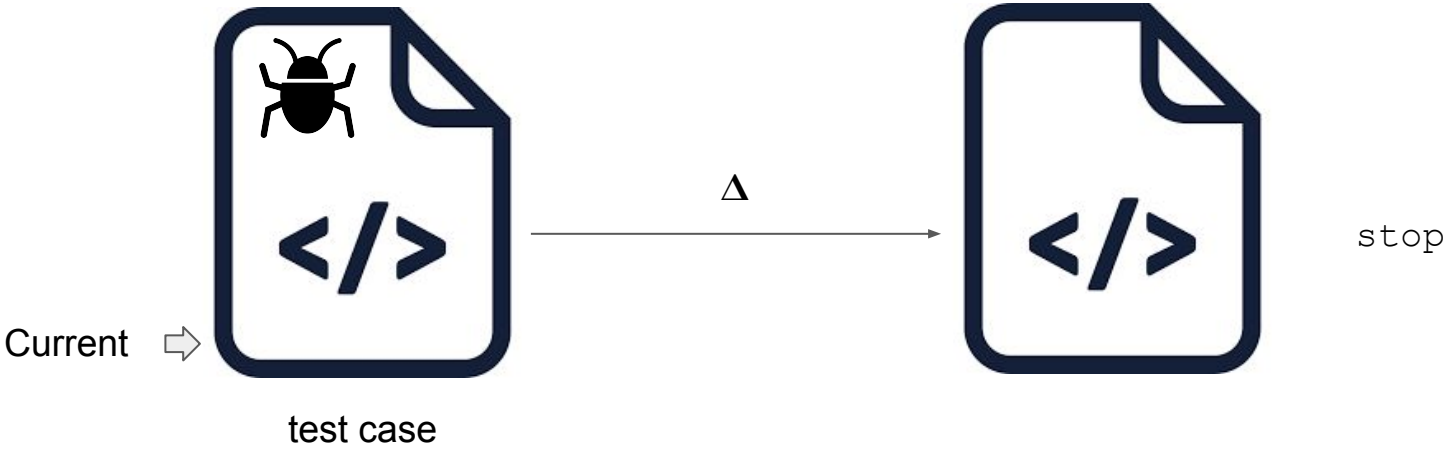
T::transform() *// alter testcase*



Creduce applies pluggable transformations until fixpoint

T : transformation

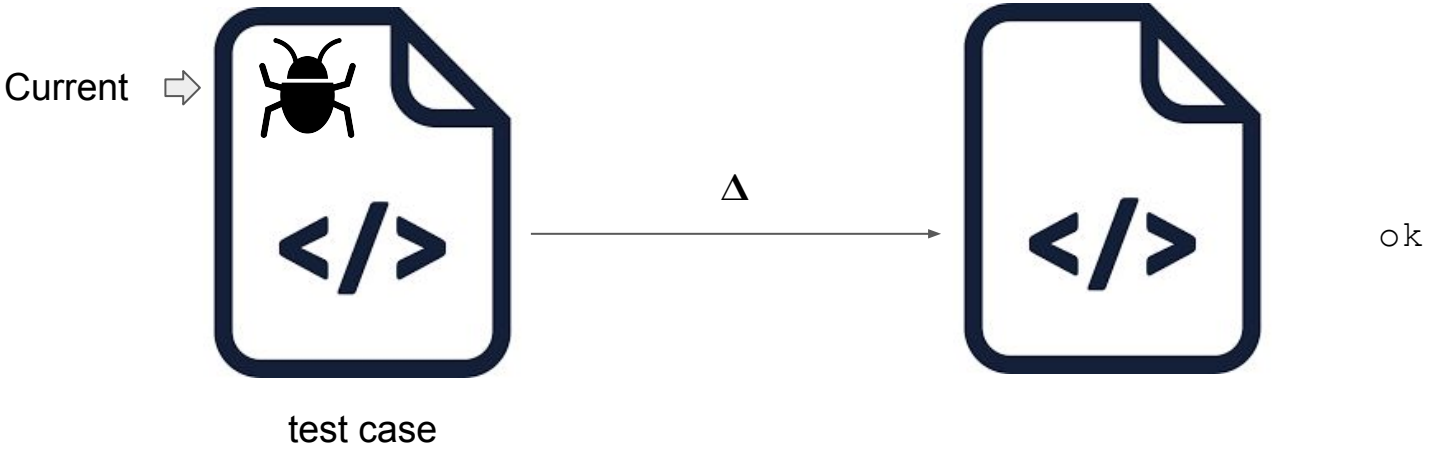
T::transform() *// alter testcase*



Creduce applies pluggable transformations until fixpoint

T : transformation

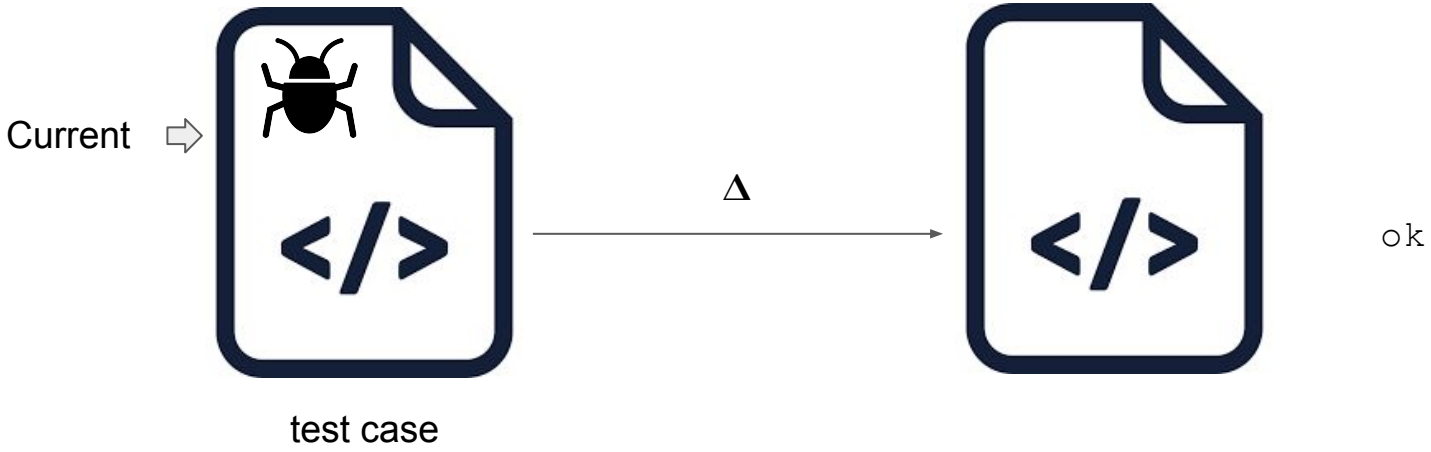
T::advance() // move on to next trial



Creduce applies pluggable transformations until fixpoint

T : transformation

T::advance() // move on to next trial



Creduce applies pluggable transformations until fixpoint

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Creduce applies pluggable transformations until fixpoint

Generalized Delta Debugging

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Creduce applies pluggable transformations until fixpoint

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Generalized Delta Debugging

- Transformation

Creduce applies pluggable transformations until fixpoint

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Generalized Delta Debugging
- Transformation
- Search

Creduce applies pluggable transformations until fixpoint

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Generalized Delta Debugging

- Transformation
- Search
- Validity check

Creduce applies pluggable transformations until fixpoint

```
current = original_test_case
while (!fixpoint) {
    foreach t in transformations {
        state = t::new ()
        while (true) {
            variant = current
            result = t::transform (variant, state)
            if (result == stop)
                break
            /* variant has behavior of interest
               and meets validity criterion? */
            if (is_successful (variant))
                current = variant
            else
                state = t::advance (current, state)
        }
    }
}
```

Listing 2. The C-Reduce algorithm

Generalized Delta Debugging

- Transformation
- Search
- Validity check
- Fitness function

Creduce outperforms existing state-of-the-art test-reduction tools

ID	Compiler	Flags	Original Size	Berkeley delta with Frama-C Size	Berkeley delta with Frama-C Time	Berkeley delta with KCC Size	Berkeley delta with KCC Time	C-Reduce with Frama-C Size	C-Reduce with Frama-C Time	C-Reduce with KCC Size	C-Reduce with KCC Time
W1	Clang 2.7	–O2	58,753	10,745	3	10,745	40	295	6	295	64
W2	GCC 3.2.0	–O3	54,301	15,153	4	15,153	30	179	5	179	42
W3	GCC 3.2.0	–O3	62,095	6,624	4	6,624	57	214	6	214	62
W4	GCC 3.3.0	–O3	54,301	15,153	4	15,153	31	176	5	176	39
W5	GCC 3.3.0	–O3	60,010	1,379	1	1,902	54	248	5	248	41
W6	GCC 3.3.0	–O3	89,036	2,414	2	2,399	47	248	5	248	46
W7	GCC 3.4.0	–O3	39,489	9,647	2	9,647	23	184	5	184	44
W8	GCC 4.0.0	–O3	42,516	1,995	5	2,550	91	134	11	134	67
W9	GCC 4.1.0	–O1	57,079	1,775	1	1,775	27	178	6	178	28
W10	GCC 4.1.0	–O1	81,067	5,789	4	4,044	113	242	9	242	215
W11	GCC 4.1.0	–O3	50,081	6,559	3	6,498	44	873	11	745	69
W12	GCC 4.1.0	–O3	57,028	11,658	3	11,658	32	202	27	202	209
W13	GCC 4.1.0	–O3	61,119	10,570	7	10,570	114	221	13	221	132
W14	GCC 4.2.0	–O0	44,078	5,208	2	5,208	21	176	5	176	32
W15	GCC 4.2.0	–O0	53,922	12,418	4	12,418	33	868	22	868	81
W16	GCC 4.2.0	–O0	56,842	15,772	7	13,585	144	971	18	971	343
W17	GCC 4.2.0	–O1	41,262	8,312	3	8,312	75	205	11	205	153
W18	GCC 4.3.0	–O0	45,298	7,816	4	7,816	63	196	7	196	79
W19	GCC 4.3.0	–O0	55,727	5,975	4	5,975	190	182	9	182	119
W20	GCC 4.3.0	–O2	64,349	10,233	9	10,233	88	205	10	205	72
W21	GCC 4.3.0	–O2	67,227	10,396	6	10,360	209	172	6	172	134
W22	GCC 4.3.0	–Os	96,273	10,689	7	10,814	119	192	12	199	663
W23	GCC 4.4.0	–O0	43,030	859	1	859	14	179	1	179	11
W24	GCC 4.4.0	–O0	52,278	1,144	1	753	130	182	1	182	73
W25	GCC 4.4.0	–O0	65,597	1,108	1	1,108	46	179	1	179	14
W26	GCC 4.4.0	–O2	40,147	4,120	2	4,120	13	755	5	755	30
W27	GCC 4.4.0	–Os	86,103	3,537	2	3,537	60	245	4	245	53
W28	Intel CC 12.0.5	–O2	42,510	13,983	6	13,983	260	187	81	317	2,312
W29	Intel CC 12.0.5	–Os	38,232	9,756	5	9,756	55	162	12	173	59
W30	Intel CC 12.0.5	–fast	48,803	2,967	4	2,967	28	185	7	196	31
W31	Intel CC 12.0.5	–fast	81,103	6,881	8	6,881	252	119	10	130	268
W32	Open64 4.2.4	–O2	43,176	1,428	2	1,428	28	218	4	218	28
W33	Open64 4.2.4	–O2	43,384	6,874	3	6,874	82	207	8	207	108
W34	Open64 4.2.4	–O2	65,732	3,976	4	3,948	77	216	5	216	91
W35	Open64 4.2.4	–O2	79,971	2,512	5	2,687	118	218	6	218	127
W36	Open64 4.2.4	–O3	96,008	30,239	11	29,956	279	160	102	160	1,129
W37	Sun CC 5.11	–xO2	41,597	1,023	1	1,023	5	148	4	148	11
W38	Sun CC 5.11	–xO2	43,176	6,391	3	6,391	54	144	8	144	86
W39	Sun CC 5.11	–xO2	43,384	7,090	3	7,090	77	145	7	145	113
W40	Sun CC 5.11	–xO2	69,657	6,025	4	6,025	108	204	13	204	143
W41	Sun CC 5.11	–xO2	70,793	1,322	1	1,322	18	145	3	145	21
		Mean	58,208	7,256	4	7,174	82	258	12	259	182
		Median	55,727	6,559	4	6,498	57	192	7	199	72

Questions