EPIC
Institute of Technology
Powered by epam

Languages

Course: Systems Architecture

Lecturer: Gleb Lobanov

March 9, 2024

EPIC
Institute of Technology
Powered by epam

Contents

EPIC
Institute of Technology
Powered by epam

01

Common knowledge

In this part of lecture, we will discuss some basic things about computers

EPIC Institute of Technology
Powered by epam

Bit operations

we have two unsigned chars a and b,

a = 11 and b = 21

11 and 21 = 1011 and 10101 = 1 =1

11 or 21 = 1011 or 10101 = 11111 = 31

11 xor 21 = 1011 xor 10101 = 11110 = 30

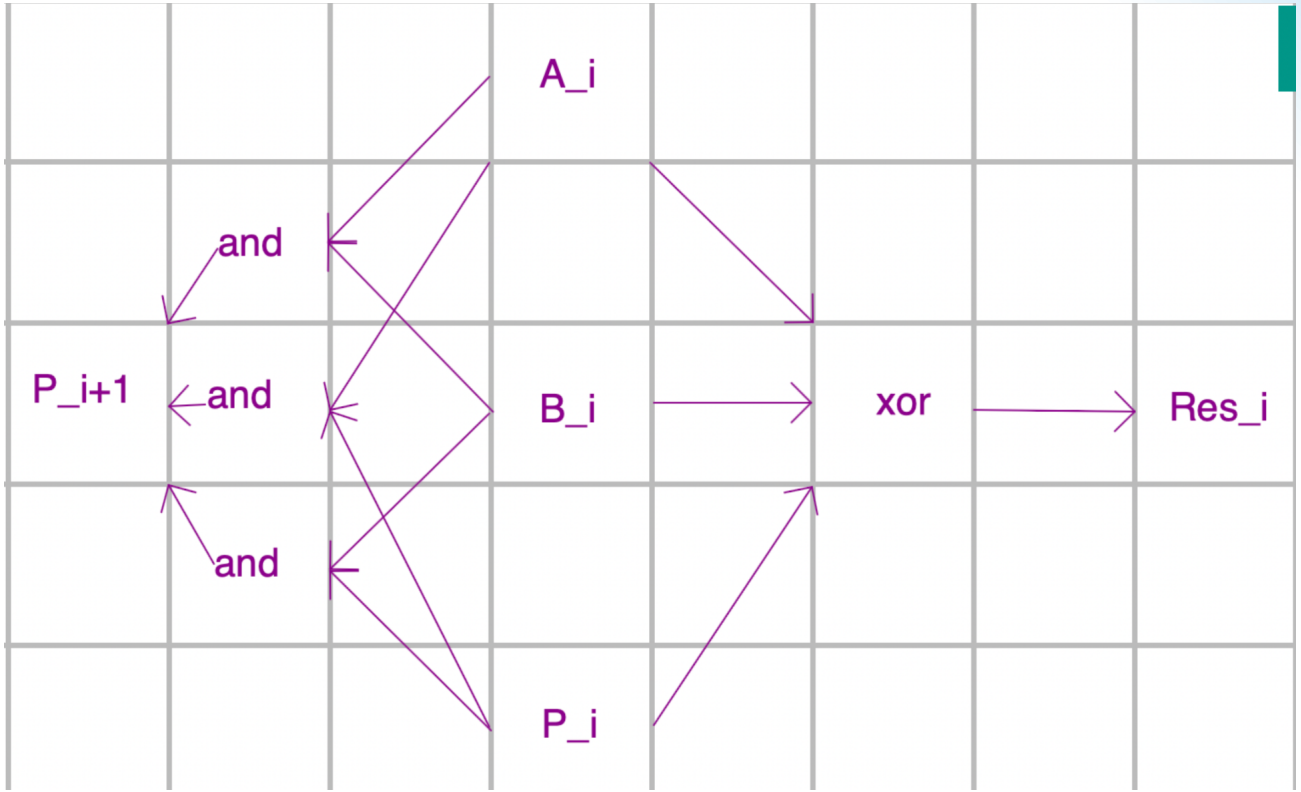
not 11 = not 1011 = 11110100 = 244

11 >> 2 = 1011 >> 2 = 10 = 2

11 << 2 = 1011 << 2 = 101100 = 44

EPIC
Institute of Technology
Powered by epam

Adder



EPIC
Institute of Technology
Powered by epam

Adder

7 + 2 = 111 + 010,

A_1 = 1, B_1 = 0, P_1 = 0 => Res_1 = 1, P_2 = 0

A_2 = 1, B_2 = 1, P_2 = 0 => Res_2 = 0, P_3 = 1

A_3 = 1, B_3 = 0, P_3 = 1 => Res_3 = 0, P_4 = 1

A_4 = 0, B_4 = 0, P_4 = 1 => Res_4 = 1, P_5 = 0

Res = 1001 = 9

EPIC
Institute of Technology
Powered by epam

Numbers

50 = 110010 (1 * 32 + 1 * 16 + 1 * 2)

- unsigned - [0; 2^n-1], n - the number of bits.
- Let's assume that char = byte = 8 bits.
- We will enumerate the bits in a byte from 0 (least significant) to 7 (most significant).

EPIC
Institute of Technology
Powered by <epam>

Negative numbers

- sign-magnitude
- one’s complement
- two’s complement

EPIC
Institute of Technology
Powered by <epam>

Sign-magnitude

0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	-0
9	1001	-1
10	1010	-2
11	1011	-3
12	1100	-4
13	1101	-5
14	1110	-6
15	1111	-7

EPIC
Institute of Technology
Powered by <epam>

One’s complement

0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	-7
9	1001	-6
10	1010	-5
11	1011	-4
12	1100	-3
13	1101	-2
14	1110	-1
15	1111	-0

EPIC
Institute of Technology
Powered by <epam>

Two’s complement

0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	-8
9	1001	-7
10	1010	-6
11	1011	-5
12	1100	-4
13	1101	-3
14	1110	-2
15	1111	-1

EPIC
Institute of Technology
Powered by epam

Two’s complement advances

1011 + 0001 = 1100

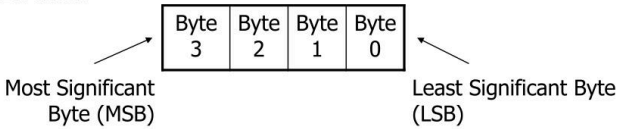
signed: -5 + 1 = -4

unsigned: 11 + 1 = 12

Little endian/Big endian

Endian-ness

- Byte Ordering for Little Endian vs. Big Endian



Memory Address	+0	+1	+2	+3	
Big Endian	Byte 3	Byte 2	Byte 1	Byte 0	MSB in the lowest (first) memory address
Little Endian	Byte 0	Byte 1	Byte 2	Byte 3	LSB in the lowest (first) memory address

- example on 8-bit unsigned
- binary - 10100001
- LE - 133
- BE - 161

EPIC
Institute of Technology
Powered by epam

02

Paradigms

This topic is about programming paradigms.

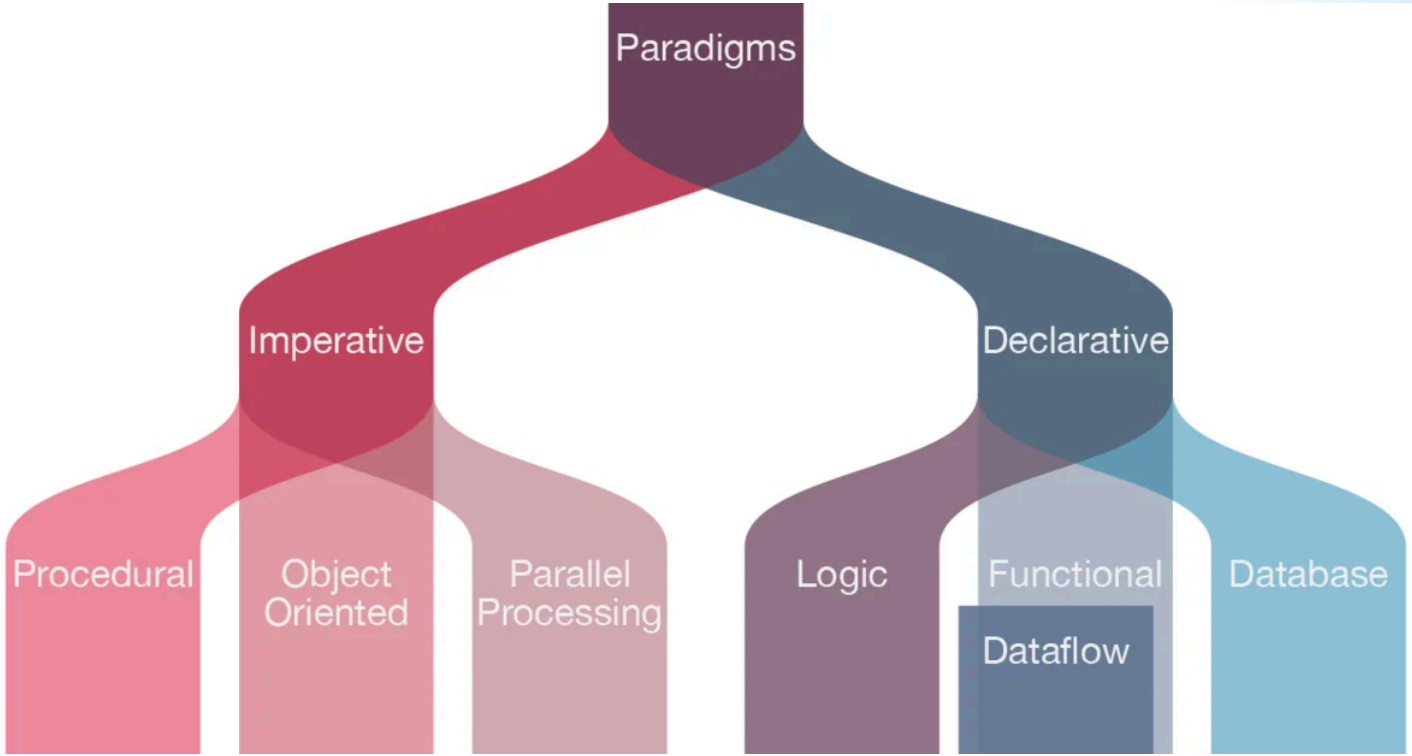
EPIC
Institute of Technology
Powered by epam

Disclaimer

Programming language is a tool that depends on personal preferences and goals of the programmer. There is no one language that fits all tasks. Each language has its advantages and disadvantages, and the choice of programming language should be based on a careful consideration of the project requirements, the programmer's level of experience, and available resources. All arguments in favor of or against a particular programming language can be subjective and dependent on the author's personal experience. Therefore, it is important to remember that the choice of programming language is an individual one, and each programmer has the right to choose what suits them and their project best.

EPIC
Institute of Technology
Powered by <epam>

Paradigms



EPIC Institute of Technology
Powered by <epam>

Imperative

```cpp
#include <iostream>

using namespace std;

int main() {
    int a[3] = {3, 1, 2};
    for (int i = 0; i < 3; i++) {
        for (int j = i; j < 3; j++) {
            if (a[i] > a[j]) {
                swap(a[i], a[j]);
            }
        }
    }

    for (int i = 0; i < 3; i++) {
        cout << a[i] << " ";
    }
    return 0;
}
```

Success #stdin #stdout 0.01s 5280KB

stdin

```
Standard input is empty
```

stdout

```
1 2 3
```

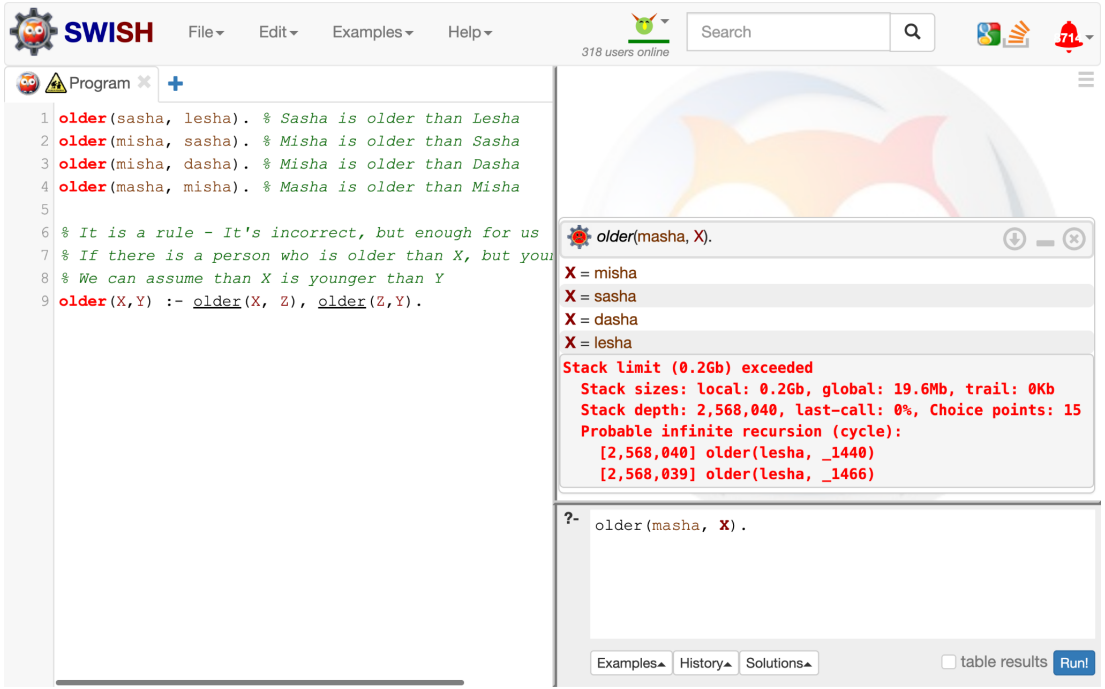
https://ideone.com/00WM3j

Full plan how to get a result

C++/Java/C/C#

EPIC
Institute of Technology
Powered by <epam>

Declarative



What we have and what do we want?

SQL/Prolog

EPIC
Institute of Technology
Powered by epam

Functional

```haskell
quicksort1 :: (Ord a) => [a] -> [a]
quicksort1 [] = []
quicksort1 (x:xs) =
    let smallerSorted = quicksort1 [a | a <- xs, a <= x]
        biggerSorted = quicksort1[a | a <- xs, a > x]
    in smallerSorted ++ [x] ++ biggerSorted

main = do
    putStrLn $ show (quicksort1 [3, 4, 2, 1, 6, 7, 4, 3])
```

Success #stdin #stdout 0.01s 5304KB

stdin

```
Standard input is empty
```

stdout

```
[1,2,3,3,4,4,6,7]
```

https://ideone.com/H912Ey

Everything is Functions

Haskell/Erlang

EPIC
Institute of Technology
Powered by epam

Paradigms Union

«In computer science, one sees several such communities, each speaking its own language and using its own paradigms. In fact, programming languages typically encourage use of some paradigms and discourage others. There are well defined schools of Lisp programming, APL programming, Algol programming, and so on. Some regard data flow, and some control flow, as the primary structural information about a program. Recursion and iteration, copying and sharing of data structures, call by name and call by value, all have adherents. »

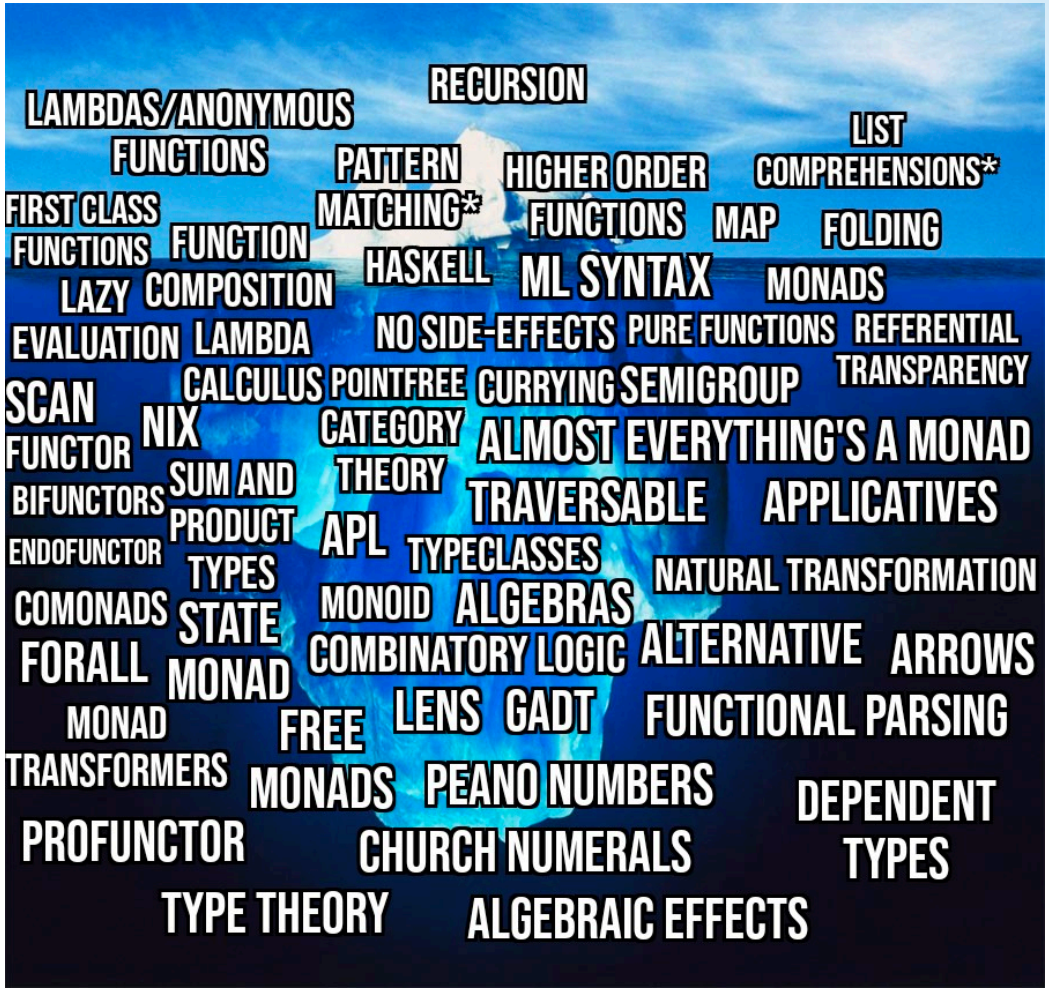
Robert W. Floyd. 1979

EPIC
Institute of Technology
Powered by epam

Welcome to the functional club

- https://github.com/rust-lang/book - Sacral rules of rust
- http://learnyouahaskell.com/chapters - More functional programming for me

EPIC
Institute of Technology
Powered by <epam>



EPIC
Institute of Technology
Powered by epam

03

C/C++

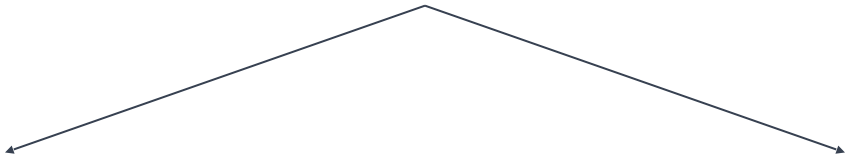
Basic stuff about C++

EPIC
Institute of Technology
Powered by epam

C/C++

- OOP (Object-Oriented Programming)
- Function Overloading
- Exceptions
- Namespaces
- References and Pointers
- Standard Library Algorithms
- Working with Strings
- Compatibility with C
- Virtual Functions
- Templates

Standards



C: C11 - 2011

C++: C++20 - 2020

EPIC
Institute of Technology
Powered by <epam>

int/byte/char

Тип	Atmel AVR	32-bit	Win64	64-bit
`char`	1	1	1	1
`short`	2	2	2	2
`int`	2	4	4	4
`long`	4	4	4	8
`long long`	-	8	8	8
`__int128 !`	-	-	-	16
`float`	4	4	4	4
`double`	4	8	8	8
`long double`	4	8 или 12	8	16
`void *`	2	4	8	8

EPIC Institute of Technology
Powered by epam

Limits of types

```cpp
#include <limits>
#include <iostream>

using namespace std;

int main()
{
    cout << numeric_limits<int>::max() << " " << numeric_limits<int>::min();
    return 0;
}
```

Success #stdin #stdout 0s 5300KB

stdin

```
Standard input is empty
```

stdout

```
2147483647 -2147483648
```

https://ideone.com/pQlGq9

```cpp
#include <iostream>
#include <stdint.h>

using namespace std;

int main() {
    int a = -11;
    long long b = a;
    long long c = -11;
    int d = c;
    cout << b << " " << d;
    return 0;
}
```

Success #stdin #stdout 0.01s 5308KB

stdin

```
Standard input is empty
```

stdout

```
-11 -11
```

https://ideone.com/lM5t6B

EPIC Institute of Technology Powered by epam

Extra info about types

sizeof(char) <= sizeof(short) <= sizeof(int) <= sizeof(long) <= sizeof(long long)

sizeof(float) <= sizeof(double) <= sizeof(long double)

sizeof(signed T) = sizeof(unsigned T)

Transforming from signed T to unsigned T and vice versa preserves the representation

EPIC Institute of Technology
Powered by <epam>

Bit operations c++

```cpp
#include <iostream>

using namespace std;

int main() {
    unsigned char a = 11, b = 21;
    unsigned char c = ~a;
    cout << (a & b) << " " << (a | b) << " " << (a ^ b)
    << " " << int(c) << " " << (a >> 2) << " " << (a << 2);
    return 0;
}
```

Success #stdin #stdout 0s 5300KB

stdin

```
Standard input is empty
```

stdout

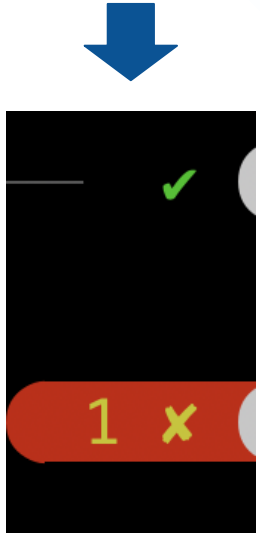
```
1 31 30 244 2 44
```

https://ideone.com/Mixh0M

EPIC
Institute of Technology
Powered by <epam>

```cpp
#include <iostream>

using namespace std;

int main() {
    return 1;
}
```

Return value



EPIC
Institute of Technology
Powered by <epam>

Pointers

```
5.  void increment (int *p) {
6.      (*p)++;
7.  }
8.
9.  int main() {
10.     int x = 10;
11.     int *pointer_x = &x;
12.     cout << *pointer_x << " ";
13.     increment(pointer_x);
14.     cout << *pointer_x << " ";
15.
16.     int *pointer_y = new int;
17.     *pointer_y = 10;
18.     cout << *pointer_y << " ";
19.     delete pointer_y;
20.
21.     int *arr = new int[5];
22.     for (int i = 0; i < 5; i++) {
23.         arr[i] = i;
24.     }
25.
26.     for (int i = 0; i < 5; i++) {
27.         cout << arr[i] << " ";
28.     }
29.
30.     delete[] arr;
31.     return 0;
32. }
```

Success #stdin #stdout 0.01s 5280KB

stdin

```
Standard input is empty
```

stdout

```
10 11 10 0 1 2 3 4
```

https://ideone.com/AxZt87

EPIC
Institute of Technology
Powered by epam

Operator ->

```cpp
#include <iostream>

using namespace std;

struct Item {
    int value;
    Item* item;
};

int main() {
    Item* f = new Item{1};
    Item* s = new Item{2};

    f->item = s; // (*f).item = s

    cout << (*(*f).item).value << " " << f->item->value;
    return 0;
}
```

Success #stdin #stdout 0s 5304KB

stdin

```
Standard input is empty
```

stdout

```
2 2
```

https://ideone.com/zMjNJ7

EPIC
Institute of Technology
Powered by epam

UB

Examples of UB:

- Dereferencing a null pointer;
- Going beyond the bounds of an array or buffer;
- Using uninitialized variables;
- Double-freeing or deleting what has already been freed or deleted;
- Operations that result in arithmetic overflow;
- Violating rules of input-output streams;
- Some multithreading errors such as race conditions..
- Violation of requirements for function arguments and templates in STL

EPIC Institute of Technology
Powered by epam

UB example 1

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int a[4];
5.
6.  bool exists_wrong(int v) {
7.      for (int i = 0; i <= 4; i++) {
8.          if (a[i] == v) {
9.              return true;
10.         }
11.     }
12.     return false;
13. }
14.
15. bool exists(int v) {
16.     return true;
17. }
```

https://ideone.com/9cgmr6

EPIC Institute of Technology
Powered by epam

UB example 2

```
1.  int fermat(){
2.      const int MAX = 1000;
3.      int a = 1, b = 1, c = 1;
4.      while (1) {
5.          if (((a * a * a) == ((b * b * b) + (c * c * c)))) {
6.              return 1;
7.          }
8.          a++;
9.          if (a > MAX) {
10.             a = 1;
11.             b++;
12.         }
13.         if (b > MAX) {
14.             b = 1;
15.             c++;
16.         }
17.         if (c > MAX) {
18.             c=1;
19.         }
20.     }
21.     return 0;
22. }
23.
```

https://ideone.com/1Jf5gP

EPIC
Institute of Technology
Powered by epam

Exceptions

```cpp
#include <iostream>
#include <stdexcept>

using namespace std;

int divide(int a, int b) {
    if (b == 0) {
        throw invalid_argument("division by 0");
    }
    return a / b;
}

int main () {
    try {
        divide (20, 0);
    }
    catch (invalid_argument& e) {
        cout << e.what() << endl;
        return -1;
    }
    return 0;
}
```

Runtime error #stdin #stdout 0.01s 5276KB

stdin

```
Standard input is empty
```

stdout

```
division by 0
```

https://ideone.com/Fj34Qs

EPIC Institute of Technology
Powered by <epam>

Command-Line Arguments

cla.cpp (~) - \

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
    if (argc != 2) {
        printf("You forget to enter your name.\n");
        exit(1);
    }
    printf("Hello %s", argv[1]);
    return 0;
}
~
~
```

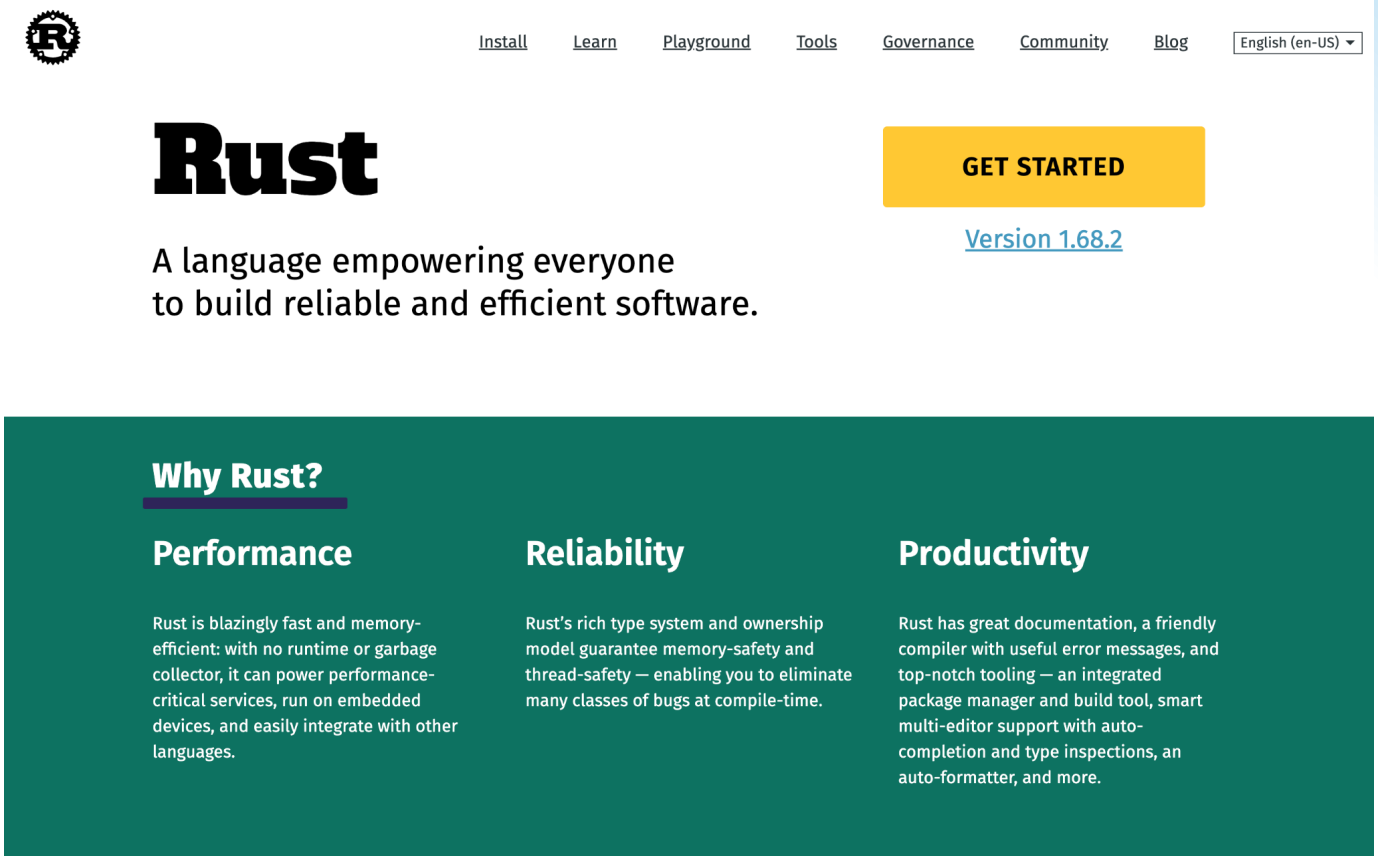
```
~
g++ cla.cpp

~
./a.out
You forget to enter your name.

~
./a.out Gleb
Hello Gleb%
```

EPIC
Institute of Technology
Powered by epam

04

Rust

Basic stuff about Rust


Install
Learn
Playground
Tools
Governance
Community
Blog
English (en-US)
Rust
GET STARTED
Version 1.68.2
A language empowering everyone
to build reliable and efficient software.
Why Rust?
Performance
Rust is blazingly fast and memory-efficient: with no runtime or garbage collector, it can power performance-critical services, run on embedded devices, and easily integrate with other languages.
Reliability
Rust's rich type system and ownership model guarantee memory-safety and thread-safety — enabling you to eliminate many classes of bugs at compile-time.
Productivity
Rust has great documentation, a friendly compiler with useful error messages, and top-notch tooling — an integrated package manager and build tool, smart multi-editor support with auto-completion and type inspections, an auto-formatter, and more.

EPIC
Institute of Technology
Powered by <epam>

Hello World

edit fork download

```
1.  use std::io::stdin;
2.  use std::io::BufRead;
3.  use std::io::BufReader;
4.
5.  fn main() {
6.      println!("Hello world")
7.  }
8.
```

Success #stdin #stdout 0.01s 5292KB

stdin

```
Standard input is empty
```

stdout

```
Hello world
```

https://ideone.com/yqiIQl

Length	Signed	Unsigned
8-bit	`i8`	`u8`
16-bit	`i16`	`u16`
32-bit	`i32`	`u32`
64-bit	`i64`	`u64`
128-bit	`i128`	`u128`
arch	`isize`	`usize`

edit fork download

```
1.  fn main () {
2.      let a = 5; // You can't change - not mut(immutable)
3.      let mut b = 5; // You can change - mut(mutable)
4.      // a = 3; no, it's prohibited
5.      b = 3;
6.      println!("a = {}, b = {}", a, b);
7.  }
```

Success #stdin #stdout 0s 5304KB

stdin

```
Standard input is empty
```

stdout

```
a = 5, b = 3
```

https://ideone.com/HzioDx

EPIC
Institute of Technology
Powered by epam

Conditions

edit fork download

```
1. fn main() {
2.     let x = 10;
3.     if x > 5 {
4.         println! ("x > 5");
5.     }
6.     else {
7.         println!("x <= 5");
8.     }
9. }
```

Success #stdin #stdout 0s 5304KB

stdin

```
Standard input is empty
```

stdout

```
x > 5
```

https://ideone.com/RKkQqd

EPIC
Institute of Technology
Powered by <epam>

Conditions

edit fork download

```
1. fn main() {
2.     println!("{}", if 3 > 4 {3} else {2});
3. }
```

Success #stdin #stdout 0s 5308KB

stdin

```
Standard input is empty
```

stdout

```
2
```

https://ideone.com/lhqRKI

EPIC Institute of Technology
Powered by <epam>

Limits of types

edit fork download

```
1.  fn main() {
2.      println!("Max value for i32: {}", i32::max_value());
3.      println!("Min value for i32: {}", i32::min_value());
4.      println!("Max value for u64: {}", u64::max_value());
5.      println!("Min value for u64: {}", u64::min_value());
6.  }
```

Success #stdin #stdout 0.01s 5280KB

stdin

```
Standard input is empty
```

stdout

```
Max value for i32: 2147483647
Min value for i32: -2147483648
Max value for u64: 18446744073709551615
Min value for u64: 0
```

https://ideone.com/JuyJRh

EPIC Institute of Technology
Powered by <epam>

Bit operations

```rust
fn main() {
    let a = 0b1011; // 11 in binary
    let b = 21;

    println!("a & b = {:b} = {}", a & b, a & b);
    println!("a | b = {:b} = {}", a | b, a | b);
    println!("a ^ b = {:b} = {}", a ^ b, a ^ b);
    println!("a << 2 = {:b} = {}", a << 2, a << 2);
    println!("b >> 2 = {:b} = {}", b >> 2, b >> 2);
}
```

Success #stdin #stdout 0.01s 5308KB

stdin

```
Standard input is empty
```

stdout

```
a & b = 1 = 1
a | b = 11111 = 31
a ^ b = 11110 = 30
a << 2 = 101100 = 44
b >> 2 = 101 = 5
```

https://ideone.com/7JrNRO

EPIC
Institute of Technology
Powered by <epam>

Loops

edit fork download

```
1.  fn main() {
2.      let mut sum1 = 0;
3.      let mut sum2 = 0;
4.
5.      for item in (0 .. 10) {
6.          sum1 += item;
7.      }
8.
9.      let mut i = 0;
10.     while i < 10 {
11.         sum2 += i;
12.         i += 1;
13.     }
14.
15.     println!("sum1 = {}, sum2 = {}", sum1, sum2);
16. }
```

Success #stdin #stdout 0s 5304KB

stdin

```
Standard input is empty
```

stdout

```
sum1 = 45, sum2 = 45
```

https://ideone.com/HBbIpx

EPIC
Institute of Technology
Powered by <epam>

Rust struct

```rust
struct Student {
    name: String,
    age: u32,
}

impl Student {
    fn student_info(&self)-> String {
        format!("Student {} is {} years old", self.name, self.age)
    }
}

fn main() {
    let student = Student {
        name: String::from("Gleb"),
        age: 22,
    };
    println!("{}", student.student_info());
}
```

Success #stdin #stdout 0s 5308KB

comments (0)

stdin copy

```
Standard input is empty
```

stdout copy

```
Student Gleb is 22 years old
```

https://ideone.com/y5N1Q2

```rust
trait Animal {
    fn make_sound(&self);
}

struct Cat {
    name: String
}

impl Animal for Cat {
    fn make_sound(&self) {
        println!("{} says meow", self.name);
    }
}

struct Dog {
    name: String
}

impl Animal for Dog {
    fn make_sound(&self) {
        println!("{} says woof", self.name);
    }
}

fn main () {
    let dog = Dog { name: String::from("Fido") };
    dog.make_sound();
    let cat = Cat { name: String::from("Whiskers") };
    cat.make_sound();
}
```

Success #stdin #stdout 0s 5268KB

stdin

```
Standard input is empty
```

stdout

```
Fido says woof
Whiskers says meow
```

https://ideone.com/F5mpDZ

```rust
struct Student {
    name: String,
    age: Option<u32>
}

impl Student {
    fn student_info(&self) -> String {
        match self.age {
            None => format!("We don't know student's age"),
            Some(age) => format!("Student {} is {} years old",
                                 self.name, age),
        }
    }
}

fn main() {
    let student1 = Student {
        name: String::from("Gleb"),
        age: Some(22)
    };
    let student2 = Student {
        name: String::from("Gleb"),
        age: None
    };
    println! ("{}", student1.student_info());
    println!("{}", student2.student_info());
}

```

stdout

```
Student Gleb is 22 years old
We don't know student's age
```

https://ideone.com/7uebLp

EPIC
Institute of Technology
Powered by <epam>

Enum

```rust
enum Direction {
    Up,
    Down
}

enum ResultCourse {
    Mark(u8),
    Fail(String)
}

fn main() {
    let direction = Direction::Up;
    match direction {
        Direction::Up => println!("Going up!"),
        Direction::Down => println!("Going down!"),
    }

    let resultStudent = ResultCourse::Fail("Miss exam".to_string());
}
```

Success #stdin #stdout 0.01s 5280KB

stdin

```
Standard input is empty
```

stdout

```
Going up!
```

https://ideone.com/VMYTVQ

EPIC
Institute of Technology
Powered by <epam>

```rust
fn divide(a: i32, b: i32) -> Option<i32> {
    if b == 0 {
        None
    } else {
        Some(a / b)
    }
}

fn main() {
    match divide(3, 0) {
        Some(result) => println!("{}", result),
        None => println!("Division by zero!"),
    }

    match divide(3, 2) {
        Some(result) => println!("{}", result),
        None => println!("Division by zero!"),
    }
}
```

Success #stdin #stdout 0.01s 5304KB

stdin

```
Standard input is empty
```

stdout

```
Division by zero!
1
```

https://ideone.com/sLaO9O

EPIC
Institute of Technology
Powered by epam

```rust
fn divide(a: i32, b: i32) -> Result<i32, String> {
    if b == 0 {
        Err(String::from("Division by zero"))
    } else {
        Ok(a / b)
    }
}

fn main() {
    match divide(3, 0) {
        Ok(result) => println!("{}", result),
        Err(e) => println!("Error: {}", e),
    }

    match divide(3, 2) {
        Ok(result) => println!("{}", result),
        Err(e) => println!("Error: {}", e),
    }
}
```

Success #stdin #stdout 0.01s 5280KB

stdin

```
Standard input is empty
```

stdout

```
Error: Division by zero
1
```

https://ideone.com/9512b8

EPIC
Institute of Technology
Powered by <epam>

Sign ?
in Rust

```rust
fn divide(a: i32, b: i32) -> Result<i32, String> {
    if b == 0 {
        Err(String::from("Division by zero"))
    } else {
        Ok(a / b)
    }
}

fn main() -> Result<(), String> {
    let result = divide(3, 0)?;
    println!("{}", result);
    Ok(())
}
```

Runtime error #stdin #stdout #stderr 0.01s 5304KB

stdin

```
Standard input is empty
```

stdout

```
Standard output is empty
```

stderr

```
Error: "Division by zero"
```

https://ideone.com/PN8JuZ

EPIC
Institute of Technology
Powered by <epam>

The ownership system

```
1.  fn main() {
2.      let s: String = String::from("Example");
3.      let t: String = s;
4.      println!("t = {}", t);
5.      println!("s = {}", s);
6.  }
7.
```

Compilation error #stdin compilation error #stdout 0s 0KB comments (0)

stdin copy

```
Standard input is empty
```

compilation info

```
error[E0382]: use of moved value: `s`
 --> prog.rs:5:24
  |
3 |     let t: String = s;
  |         - value moved here
4 |     println!("t = {}", t);
5 |     println!("s = {}", s);
  |                        ^ value used here after move
  |
  = note: move occurs because `s` has type `std::string::String`, which does not implement the `Co
py` trait

error: aborting due to previous error

For more information about this error, try `rustc --explain E0382`.
```

https://ideone.com/tC1EOb

EPIC
Institute of Technology
Powered by epam

Memory leak problem

https://doc.rust-lang.org/std/mem/fn.forget.html - the simplest example


IS THIS DRESS
BLUE AND BLACK
OR WHITE & GOLD?

```
1  // https://doc.rust-lang.org/std/index.html
2  use std::fs::File;
3  use std::mem;
4  fn main() {
5      //let file = File::open("file.txt");
6  }
7
```

EPIC Institute of Technology
Powered by <epam>

Dereferencing a pointer

```rust
// Dereferencing a pointer
fn main() {
    let mut x = 10;
    let ptr = &mut x as *mut i32;
    unsafe {
        *ptr = 20;
    }
}

// unsafe function
unsafe fn my_func() -> i32 {
    // ...
    1
}

// unsafe interface
unsafe trait MyTrait {
    // ...
}
```

Success #stdin #stdout 0s 5308KB

stdin

```
Standard input is empty
```

stdout

```
Standard output is empty
```

https://ideone.com/iKyA3N

```rust
// Dereferencing a pointer
fn main() {
    let mut x = 10;
    let ptr = &mut x as *mut i32;
    *ptr = 20;
}

// unsafe function
unsafe fn my_func() -> i32 {
    // ...
    1
}

// unsafe interface
unsafe trait MyTrait {
    // ...
}
```

Compilation error #stdin compilation error #stdout 0s 0KB comments (0)

stdin copy

```
Standard input is empty
```

compilation info

```
error[E0133]: dereference of raw pointer is unsafe and requires unsafe function or block
 --> prog.rs:5:5
  |
5 |     *ptr = 20;
  |     ^^^^^^^^^ dereference of raw pointer
  |
  = note: raw pointers may be NULL, dangling or unaligned; they can violate aliasing rules and cau
se data races: all of these are undefined behavior

error: aborting due to previous error

For more information about this error, try `rustc --explain E0133`.
```

EPIC Institute of Technology
Powered by <epam>

Command-Line Arguments

```rust
use std::env;

fn main() {
    let args: Vec<String> = env::args().collect();
    let argc = args.len();
    let argv = &args;

    println!("argc: {}", argc);
    println!("argv: {:?}", argv);
}
```

Success #stdin #stdout 0.01s 5172KB

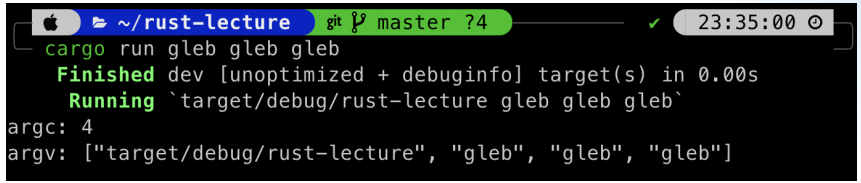
stdin

```
Standard input is empty
```

stdout

```
argc: 1
argv: ["./prog"]
```

https://ideone.com/TteWF5



EPIC Institute of Technology
Powered by <epam>

Lambda Functions

```
1.  fn main() {
2.      let add = |x, y| x + y;
3.      let result = add(5, 7);
4.      println!("{}", result);
5.  }
```

Success #stdin #stdout 0.01s 5304KB

stdin

```
Standard input is empty
```

stdout

```
12
```

https://ideone.com/JmAGM3

EPIC Institute of Technology
Powered by <epam>

Lambda Functions

```
1.  fn main() {
2.      let meters = 1000; // meters in kilometer
3.      let distanceToMetres = |x| x * meters;
4.      let result = distanceToMetres(5);
5.      println!("{}", result);
6.  }
7.
```

Success #stdin #stdout 0.01s 5276KB

stdin

```
Standard input is empty
```

stdout

```
5000
```

https://ideone.com/msUYLx

EPIC
Institute of Technology
Powered by epam

Lambda Functions

```rust
fn calc<F>(f: F, x: Vec<i32>, neutral: i32) -> i32
where
    F: Fn(i32, i32) -> i32
{
    let mut res = neutral;
    for elem in x {
        res = f(res, elem);
    }
    res
}

fn main() {
    let sum = |a, b| a + b;
    let mult = |a, b| a * b;

    let result1 = calc(sum, vec![1, 2, 3, 4, 5, 6], 0);
    let result2 = calc(mult, vec![1, 2, 3, 4, 5, 6], 1);

    println!("sum = {}, mult = {}", result1, result2);
}
```

Success #stdin #stdout 0.01s 5284KB

stdin

```
Standard input is empty
```

stdout

```
sum = 21, mult = 720
```

https://ideone.com/8umXR8

EPIC
Institute of Technology
Powered by epam

Iterators

```
1. fn main() {
2.     let vec = vec![1, 2, 3, 4, 5, 6];
3.     let collected: Vec<i32> = vec.iter()
4.         .map(|x| *x + 1)
5.         .filter(|x| *x < 4)
6.         .collect();
7.
8.     println!("{:?}", collected);
9. }
```

Success #stdin #stdout 0s 5308KB

stdin

```
Standard input is empty
```

stdout

```
[2, 3]
```

https://ideone.com/L7B7VX

EPIC
Institute of Technology
Powered by epam

Macro

```
1.  macro_rules! hello {
2.      () => {
3.          println!("Hello, world!");
4.      };
5.  }
6.
7.  macro_rules! sum {
8.      ($($x:expr),*) => {
9.          {
10.             let mut sum = 0;
11.             $(sum += $x;)*
12.             sum
13.         }
14.     };
15. }
16.
17. fn main() {
18.     hello!();
19.     println!("{}", sum!(1, 2, 3, 4));
20. }
```

Success #stdin #stdout 0s 5284KB

stdin

```
Standard input is empty
```

stdout

```
Hello, world!
10
```

https://ideone.com/Lxs7VK

CE

```rust
use std::io::stdin;
use std::io::BufRead;
use std::io::BufReader;

fn main() {
    println!("{} is {} in binary", 2/*, 10*/);
    println!("{0} is {0:b} in binary", "3")
}

```

Compilation error #stdin compilation error #stdout 0s 0KB comments (0)

stdin copy

```
Standard input is empty
```

compilation info

```
error: 2 positional arguments in format string, but there is 1 argument
 --> prog.rs:6:15
  |
6 |     println!("{} is {} in binary", 2/*, 10*/);
  |               ^^    ^^
```

https://ideone.com/iDGHzs

EPIC
Institute of Technology
Powered by epam

RE

```
1.  #include <iostream>
2.
3.  int main() {
4.      printf("%s", 2);
5.      return 0;
6.  }
```

Runtime error #stdin #stdout 0.04s 5292KB

stdin

```
Standard input is empty
```

stdout

```
Standard output is empty
```

https://ideone.com/8cqpgI

EPIC
Institute of Technology
Powered by <epam>

CE

```cpp
#include <type_traits>
#include <cstdio>

#define PRINTF_INT(expr) static_assert(std::is_integral<decltype(expr)>::value, "Not an integer"); printf("%d", expr);

int main() {
    int value = 5;
    float value_f = 5.1;

    PRINTF_INT(value);   // This will compile
    PRINTF_INT(value_f); // This will cause a compile-time error

    return 0;
}
```

Compilation error #stdin compilation error #stdout 0s 0KB

comments (0)

stdin

copy

```
Standard input is empty
```

compilation info

```
prog.cpp: In function ‘int main()’:
prog.cpp:4:40: error: static assertion failed: Not an integer
 #define PRINTF_INT(expr) static_assert(std::is_integral<decltype(expr)>::value, "Not an intege
r"); printf("%d", expr);
                                        ^~~
prog.cpp:11:5: note: in expansion of macro ‘PRINTF_INT’
     PRINTF_INT(value_f); // This will cause a compile-time error
     ^~~~~~~~~~
```

https://ideone.com/tJUg2V

EPIC
Institute of Technology
Powered by epam

05

Comparison

This section is dedicated to comparing C++ and Rust.

EPIC
Institute of Technology
Powered by epam

Rust vs C++

c++:

1) many books, topics, editorials
2) faster

rust:

1) safety from box
2) strict typization
3) something like garbage collector
4) human-readable errors
5) newer language

EPIC
Institute of Technology

```rust
fn hash_code(x: String) -> i32 {
    let mut h = 13i32;
    for i in 0..3 {
        h += h * 27752 + x.as_bytes()[i] as i32;
    }
    if h < 0 {
        h += i32::max_value();
    }
    return h;
}

fn main() {
    let h = hash_code("bye".to_string());
    println!("hash: {}", h);
}
```

Success #stdin #stdout 0s 5300KB

stdin

```
Standard input is empty
```

stdout

```
hash: 1798783020
```

https://ideone.com/yBjjWS

```cpp
#include <iostream>
using namespace std;

unsigned MAX_INT = 2147483647;

int hash_code(std::string x) {
    int h = 13;
    for (unsigned i = 0; i < 3; i++) {
        h += h * 27752 + x[i];
    }
    if (h < 0) h += MAX_INT;
    return h;
}

int main() {
    cout << hash_code("bye");
}
```

Success #stdin #stdout 0.01s 5308KB

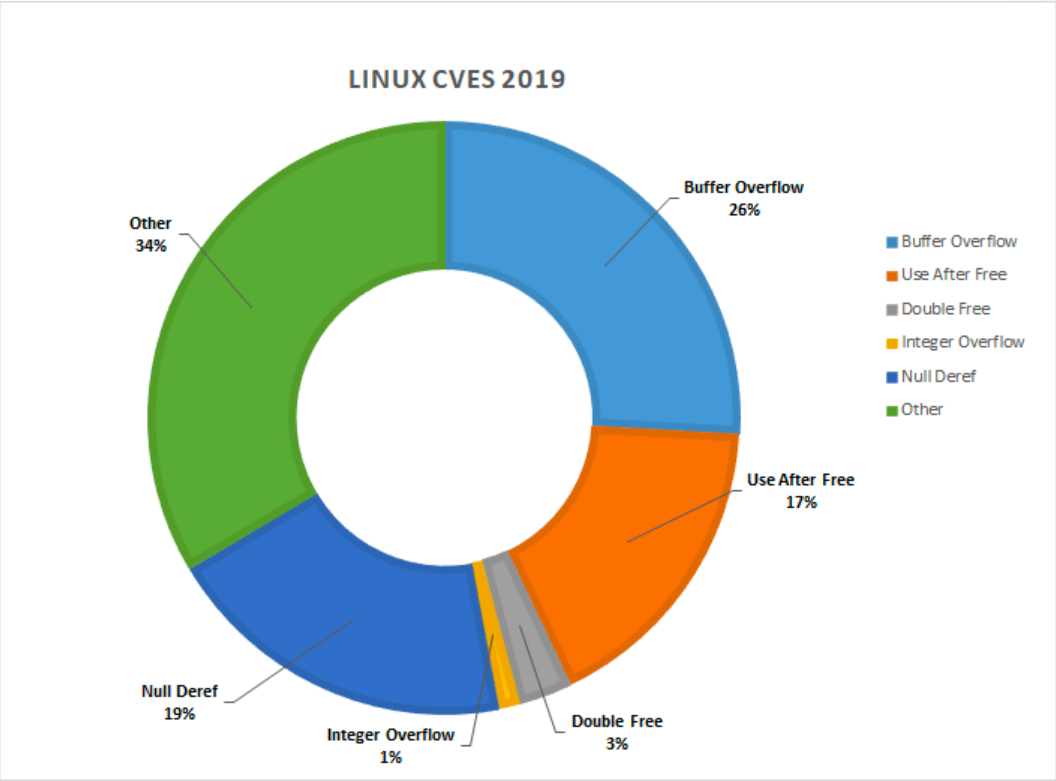
stdin

```
Standard input is empty
```

stdout

```
2147483647
```

https://ideone.com/WDniDp


LINUX CVES 2019
Buffer Overflow
26%
Use After Free
17%
Double Free
3%
Integer Overflow
1%
Null Deref
19%
Other
34%
Buffer Overflow
Use After Free
Double Free
Integer Overflow
Null Deref
Other

EPIC
Institute of Technology
Powered by epam

Benchmarks

https://programming-language-benchmarks.vercel.app/cpp-vs-rust

Torvalds had this to say about C++ in an e-mail exchange in 2007:

> *"C++ is a horrible language. It's made more horrible by the fact that a lot of substandard programmers use it, to the point where it's much much easier to generate total and utter crap with it. Quite frankly, even if the choice of C were to do *nothing* but keep the C++ programmers out, that in itself would be a huge reason to use C."*
>
> *Torvalds on C++ in 2007*

"Before the Rust people get all excited," the Linux kernel creator and chief said. "Right? You know who you are. To me, it's a trial run, right? We want to have [Rust's] memory safety. So there are real technical reasons why Rust is a good idea in the kernel.

"But at the same time, it's one of those things: We tried C++ 25-plus years ago and we tried it for two weeks and then we stopped trying it. So to me, Rust is a way to try something new. And hopefully, it works out, and people have been working on it a lot, so I *really* hope it works out because otherwise they'll be bummed."

EPIC
Institute of Technology
Powered by epam

That's All Folks!