Informatik - Exercise Session
Vectors and References

Tip: The Ternary Operator

There are several ways to shorten this snippet:

```
if (condition) {
    return a;
} else {
    return b;
}
```

Variant 1:

```
if (condition) return a; else return b;
```

The ternary operator:

```
return (condition) ? a : b;
```

Other ways to use this:

```
int i = (condition) ? a : b;
function1(one, (condition) ? a : b, three);
```

There are more ways to use this, but it gets confusing fast, so try not to overdo it.

References - Introduction

What is the output of the following snippet?

```
int a = 3;
int& b = a;
b = 7;
std::cout << a; // Output: ?
```

The output is 7:

Variable	Values	
a	3$\cancel{3}$	7 *output*
b	$\hookrightarrow$ a	7 $\uparrow\hookrightarrow$ a

References - Pass by Value

What is the output of the following program?

```
void foo(int i) {
    i = 5;
}

int main() {
    int i = 4;
    foo(i);
    std::cout << i;
}
```

Variable	Values
i	4 *output*

foo:

Variable	Values
i	5

~~foo:~~

~~Variable~~	~~Values~~
~~i~~	~~5~~

References - Pass by Reference

What is the output of the following program?

```
void foo(int& a) {
    a = 5;
}

int main() {
    int i = 4;
    foo(i);
    std::cout << i;
}
```

Variable	Values	
i	4~~4~~	5 *output*

foo:

Variable	Values	
a	↪ i	↑ 5↪ i

~~foo:~~

~~Variable~~	~~Values~~	
~~a~~	~~↪ i~~	~~↪ i~~

References - Applications

What applications of references come to mind?

- More than one “return value”:

```
void midnight(double a, double b, double c, double& x1, double & x2);
```

- Streams cannot be copied:

```
void output(std::ostream out, int i) { out << i; }  // error
void output(std::ostream& out, int i) { out << i; } // works
```

- Return references:

```
int& increment(int& m) { return ++m; }
int main() {
    int n = 3;
    increment(increment(n));
    std::cout << n; // 5
    return 0;
}
```

Consider the normalized floating point number system $F^*(\beta, p, e_{\min}, e_{\max})$ with $\beta = 2$, $p = 3$, $e_{\min} = -4$, $e_{\max} = 4$.

Compute the following expressions as the parentheses suggest, representing each intermediate result (and the final result) in the normalized floating point system according to the rules of computing with floating point numbers.

$(10 + 0.5) + 0.5$

	decimal		binary
	10		????? $1.01 \cdot 2^3$
$+$	0.5		????? $0.0001 \cdot 2^3$
$=$			????? $1.0101 \cdot 2^3$ $1.01 \cdot 2^3$
$+$	0.5		????? $0.0001 \cdot 2^3$
$=$	?? 10	$\leftarrow$	????? $1.01 \cdot 2^3$

$(0.5 + 0.5) + 10$

	decimal		binary
	0.5		????? $1.00 \cdot 2^{-1}$
$+$	0.5		????? $1.00 \cdot 2^{-1}$
$=$			????? $1.00 \cdot 2^0$
$+$	10		????? $1010.00 \cdot 2^0$
$=$	?? 12	$\leftarrow$	????? $1011.00 \cdot 2^0$ $1.011 \cdot 2^3$ $1.10 \cdot 2^3$