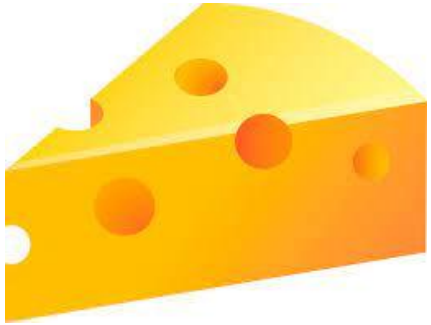
# Cheez



Final Presentation

# Team Members

Ruize Li

Jingyi Wang

Ken Xiong

Jianyang Duan

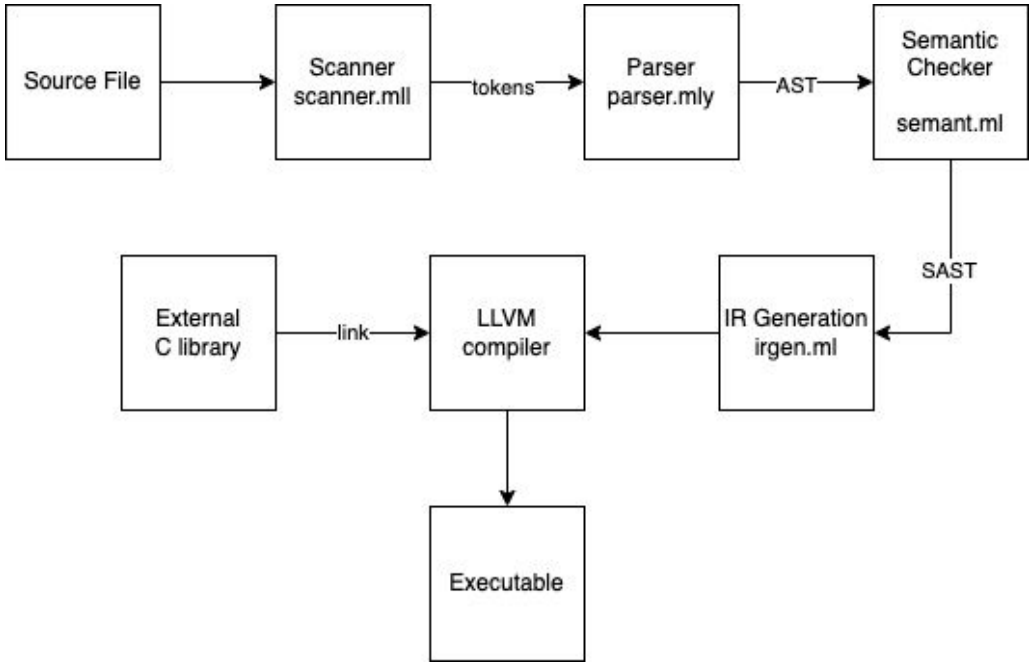
# Language Intro

Cheez is an **imperative, statically typed** language with the motivation to mimic object orientation tasks and facilitate non-cs domain people with their research area.

Cheez supports primitive data types, (eg. int, bool, float, string), basic unary, binary operation (eg. plus, minus), and two powerful non-primitive data types array and struct.

Flexibility of built-in functions.

# Architectural Design



# Language Features - Primitives

| | |
|---|---|
| Int | `int num = 1;` |
| Float | `float num = 1.5;` |
| Bool | `bool learnedocaml = true;` |
| Void | `void main() { prints('hello'); }` |
| String | `string name = 'ken';` |

# Language Features - Operators

| | | | |
|---|---|---|---|
| '+' | Plus | '<' | Less than |
| '-' | Minus | "<=" | Less than or equal to |
| '*' | Multiply | "==" | Equal to |
| '/' | Divide | "!=" | Not equal to |
| '=' | Assign | "&&" | And |
| '>' | Greater than | "\|\|" | Or |
| ">=" | Greater than or equal to | "!" | Not |

# Language Features - Array

- Array Type
  - Array supports all primitive data types such as int, bool, and float
- Array Declaration
  - Array must be declared with initial values
  - After declaration, array size is fixed and cannot be changed

```
int[] arri = [1, 2, 3];
bool[] arrb = [true, false, true];
float[] arrf = [2.3, 4.5, 1.0];
```

# Language Features - Array

- Array Access
  - Array element can be accessed via index
  - Index can be integer or variable with integer value
- Array Element Assignment
  - Array element can be re-assigned with valid value

```
int[] a = [1,2,3];
int r = 2;
int temp = a[0];
a[0] = a[r];
a[r] = temp;
```

# Language Features - Array

- Error handling
  - Array index out of bound

```
int[] arr = [1, 2, 3];
arr[4] = 5;
```

```
make all && ./cheez.native < temp.cheez > temp.out && lli
temp.out
    opam exec
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis
cheez.native
Finished, 25 targets (0 cached) in 00:00:03.
# ocamlbuild -pkgs llvm,llvm.analysis cheez.native
cc    -c -o lib.o lib.c
Fatal error: exception Failure("Index out of bound")
```

# Language Features - Array

- Error handling
  - Array elements type inconsistent

```
int[] arr = [1, 2, true];
```

```
make all && ./cheez.native < temp.cheez > temp.out && lli
temp.out
    opam exec
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis
cheez.native
Finished, 25 targets (0 cached) in 00:00:03.
# ocamlbuild -pkgs llvm,llvm.analysis cheez.native
cc    -c -o lib.o lib.c
Fatal error: exception Failure("Type inconsistent int bool")
```

# Language Features - Array

- Error handling
  - Array element assignment type mismatch

```
int[] arr = [1, 2, 3];
arr[0] = 3.7;
```

```
make all && ./cheez.native < temp.cheez > temp.out && lli
temp.out
    opam exec
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis
cheez.native
Finished, 25 targets (0 cached) in 00:00:03.
# ocamlbuild -pkgs llvm,llvm.analysis cheez.native
Fatal error: exception Failure("type mismatch")
```

# Language Features - Struct

- Struct Declaration
  - Structs must be declared ahead of the functions and the struct name must start with a capital letter.
- Variable Initialization
  - Within the struct declaration, all fields can be constructed under variable initialization without populating a specific value.

```
struct Person {
    int age;
    string name;
};
```

# Language Features - Struct

- Struct Access
  - Fields can be accessed by directly referring to their names.
- Struct Assign
  - Struct fields can be assigned  to valid values.

```
int main()
{
    Person person1;
    person1.name = 'Ken';
    person1.age = 22;
    prints(person1.name);
    printi(person1.age);

    return 0;
}
```

```
ocamlbuild -clean
Finished, 0 targets (0 cached) in 00:00:00.
00:00:00 0    (0   ) STARTING
make all && ./cheez.native < temp.cheez > temp.out && lli temp.out
    opam exec
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis cheez.native
Finished, 25 targets (0 cached) in 00:00:04.
cc    -c -o lib.o lib.c
'Ken'
22
```

# Language Features - Struct

- Error handling
  - Struct variable not found

```
struct Person{
    int age;
    string name;
};

int main()
{
    Person person1;
    person1.name = 'Ken';
    person2.age = 22;
    prints(person1.name);
    printi(person1.age);

    return 0;
}
```

```
(base) Kens-MBP:PLT_chez kenxiong$ make clean && make temp
ocamlbuild -clean
Finished, 0 targets (0 cached) in 00:00:00.
00:00:00 0    (0   ) STARTING  -------- |rm -rf *.native *.out *.o ./test/*.out
make all && ./cheez.native < temp.cheez > temp.out && lli temp.out
# opam config exec -- \
    opam exec
ocamlbuild -use-ocamlfind -pkgs llvm,llvm.analysis cheez.native
Finished, 25 targets (0 cached) in 00:00:04.
# ocamlbuild -pkgs llvm,llvm.analysis cheez.native
cc    -c -o lib.o lib.c

Fatal error: exception Failure("ID not found: person2")
```

# Language Features - Built-ins

- Print functions in printing in string, boolean, integer, float format
- String built-in library from external C library
  - Utilized the functions from string.h library of C
  - Added additional functions in C file (upper, lower, substring)
  - Linked in with llvm in codegen stage as built in functions in our language

# Testing

Testing Suite : Python script

- Unit test
- What counts as a PASS ?
- Manual analytics
- Most bugs detected during compilation

```
(base) ➜  PLT_chez git:(IR_initial) make test
python3 test.py

  ----------------------------------------
  |                                      |
  |   Welcome to the Cheez testing suite!   |
  |                                      |
  |--------------------------------------|

[+] Running test "test_6_binop_lt"...
[+] test "test_6_binop_lt" PASSED.

[+] Running test "test_5_ctrl_for"...
[+] test "test_5_ctrl_for" PASSED.

[+] Running test "test_9_complex_gcd"...
[+] test "test_9_complex_gcd" PASSED.



. . .
```

# Testing - Examples

```
int main()

{

    printi(gcd(10,100));

}



int gcd(int a, int b) {

    while (a != b)

        if (a > b) a = a - b;

        else b = b - a;

    return a;

}
```

```
[+] Running test "test_9_complex_gcd"...
[+] test "test_9_complex_gcd" PASSED.
```

```
int main()
{
    int[] arr = [5,4,3,2,1];
    int n = 5;
    // selection sort
    for (int i = 0; i < n - 1; i = i + 1)
    {
        int min = i;
        for (int j = i + 1; j < n; j = j + 1)
        {
            if (arr[j] < arr[min]) {
                min = j;
            }
        }
        // swap
        int tmp = arr[i];
        arr[i] = arr[min];
        arr[min] = tmp;
    }
    // print
    for (int k = 0; k < n; k = k + 1)
    {
        printi(arr[k]);
    }

}
```

```
[+] Running test "test_9_complex_selectionsort"...
[+] test "test_9_complex_selectionsort" PASSED.
```

# Future Work

- Multi-dimensional arrays
- Nested structs
- Type casting
- Array library functions, i.e. reverse, pop, append
- Dock container for testing

# Demo