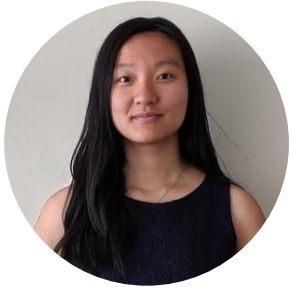
AHOD - 

Caitlyn Chen, Tiffeny Chen, Jang Hun Choi, Mara Dimofte, Christi Kim
PLT Spring 2021

Team

Caitlyn Chen
Language Guru



Tiffeny Chen
System Architect

Jang Hun Choi
System Architect

Mara Dimofte
Manager

Christi Kim
Tester

TA: Xijao Li

Motivation

- Card games come in many different forms -- standard 52-card deck to UNO, Apples, and Pokemon relying on unique decks.

- Existing card languages fail to generalize the full breadth of card games

- Allows users to easily code the gameplay and the functionality of a turn-based card game

Language Tutorial - Basic

global var

required main declaration

local var

assignment

control structure: for, while, if-else

end of scope for main

```
1   int a
2   main:
3   {
4       float b
5       b = 5.0
6       if 3<5:
7       {
8           for (a = 0 ; a < 5; a = a + 1):
9           {
10              b = b + 1.0
11              do PRINT(b)
12          }
13      }
14  }
15
```

*strongly typed language
- *types*: int, float, bool, string, void, series, player, card

*newline instead of semi colon

Language Tutorial - Advanced

action (func) CREATEPILE

```
1   when do series<Card> CREATEPILE():
2   {
3       Card card1
4       card1 = Card("R5", true)
5       return [card1]
6   }
7   main:
8   {
9       series<Card> cards
10      Card card
11      cards = do CREATEPILE()
12      card = cards[0]
13      do PRINT(card.type)
14      do PRINT(card.faceup)
15  }
16
```

created card object

returned series literal

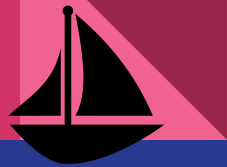
called action

accessed 0th element of series cards

card attributes: R5, true (respectively)

Features Summary

Feature	Associated Methods
Control Flow	if, else, while, for
Built in Objects	Player, Card
Series Literals	push, pop, get, size
Action Declarations	accessing params, return



Built-in Objects

Player

```
1   Player player
2   main:
3   {
4     player = Player("bob", 0)
5     do PRINT(player.name)
6     do PRINT(player.score)
7   }
8
```

Card

```
1   Card card
2   main:
3   {
4     card = Card("R5", true, 5)
5     do PRINT(card.type)
6     do PRINT(card.faceup)
7     do PRINT(card.value)
8   }
```

Series Literal

```
series<int> a
series<string> b
int i
main:
  {
    a = [2]

    a.push(1)
    do PRINT(a[1])

    a.push(3)
    do PRINT(a[2])

    a.pop()

    for  (i = 0; i < a.size(); i = i + 1):
    {
    do PRINT(a[i])
    }

    b = ["stmt1", "stmt2"]

    b.push("stmt3")
    b.push("stmt4")

    do PRINT(b[3])

  }
```

Action Declarations

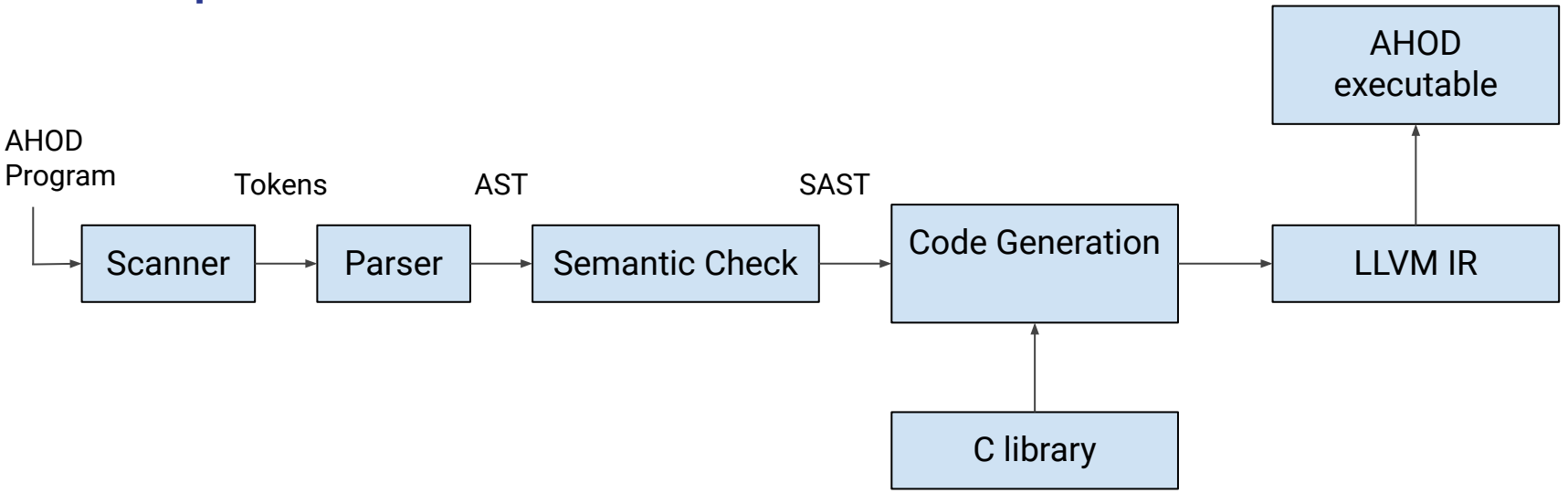
Program

```
1   when do string A(string x):
2   {
3       string z
4       z = x
5       return z
6   }
7   main:
8   {
9       string w
10      w = do A("Ahoy Matey, to AHOD")
11      do PRINT(w)
12  }
```

Output

```
1   Ahoy Matey, to AHOD
```

Compiler Architecture



Demo

Next Steps

- Create more built-in-methods:
  - string-to-int, int-to-string, user input, etc.
- Make code more concise and flexible
  - use whitespace as delimiters (indentation, newline, etc.)
- Allow users to define classes and class attributes
- More list implementations
  - slicing, remove and find certain element
- Type inferencing
- Optimizations of register allocations
- Garbage collection

Thank you for all the guidance this semester!