GASSP

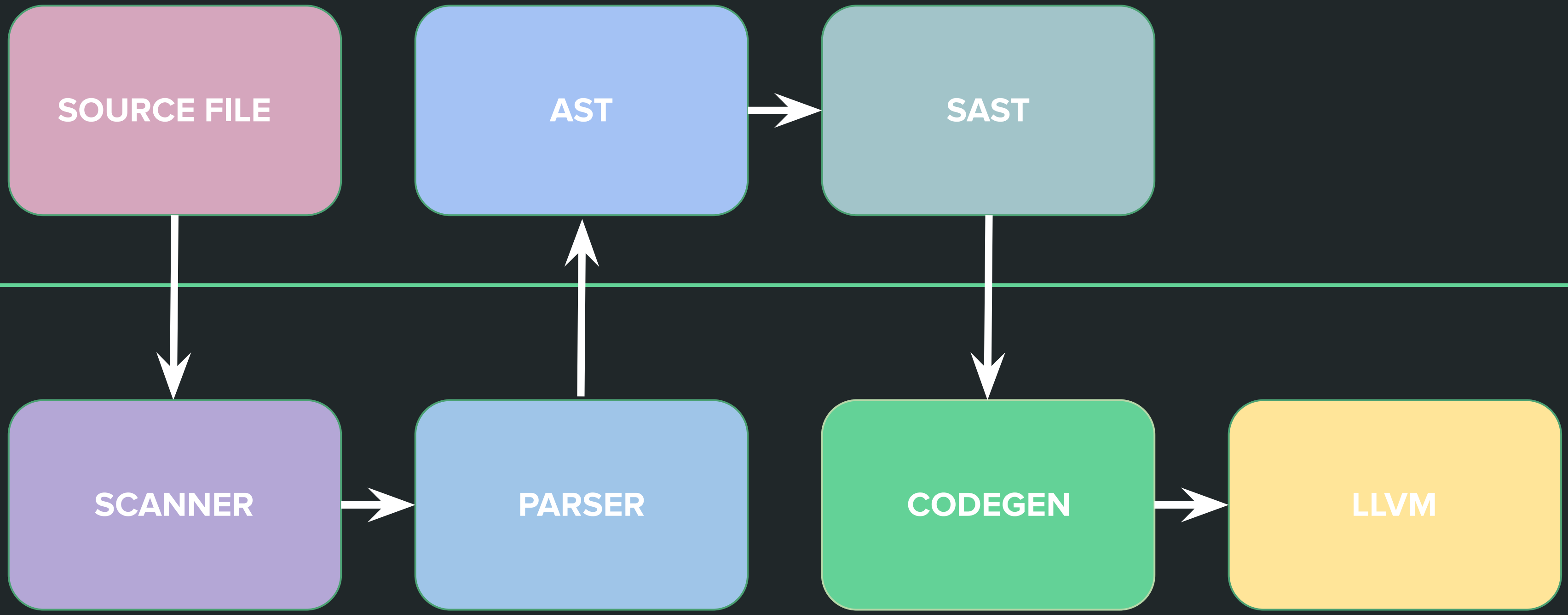
Adam Fowler	Language Guru
Patrycja Przewoźnik	Tester
Swan Htet System	System Architect
Yuanxin Yang	System Architect
Sam Weissman	Manager

Introduction to GASSP

- GASSP is a statically typed object-oriented general purpose programming language with its roots in C++ and Java

- Functionality:
  - Classes and objects
  - Break and Continue statements
  - Lots of functions → string manipulation and math
  - Support for strings and floating point numbers

System Architecture



SYNTAX

OPERATORS

Float

- ADD
- SUB
- MULT
- DIV
- EQUAL
- NEQ
- LESS
- LEQ
- GREATER
- GEQ

Boolean

- AND
- OR
- NOT

INT

- MOD
- LSHIFT
- RSHIFT
- INCR
- DECR

+ Float operators

BUILT IN TYPES

● INT	➜ 4115
● CHAR	➜ ‘g’
● BOOL	➜ true, false
● FLOAT	➜ 3.14
● STRING	➜ ‘Gassp’
● OBJECT	
● VOID	

Control flow

```
if (pred)  {
   print(“pred was true!”);
} else { // this is optional
   print(“pred was false”);
}
```

```
for(int i = 0; i < 5; i++){
     print(i);
}

while(true) {
//loop forever
}
```

Control Flow: continue & break statements

```
 prints("With continue
statement: \n")
   x = 5;
   while(x > 0)
    {
       print(x);
       prints("\n");
       x = x - 1;
       if( x == 3)
       {
        /* skip the iteration */
        x = x - 1;
        continue;
       }
    }
```

```
   prints("With break statement: \n")
   x = 5;
   while(x > 0)
    {
        print(x);
        prints("\n");
        x = x - 1;
        if( x == 3)
        {
         /* breaks the loop */
         break;
        }
    }
}
```

CLASSES

- Comprised of fields and methods
- Supports visibility modifiers
- Syntax:

```
<class access_modifier> class <class name> {
    <field declarations>
    <constructors declarations>
    <method declarations>
}
```

OBJECTS

- FIELDS
  - Primitive types or other object classes
  - Access & modification through DOT (.)

- METHODS
  - Defined inside a class
  - Called through DOT (.)
    - `<classname>.<attrname>`
    - `Ex. Course.courseNumber`
  - Inputs, outputs → primitives, objects
  - Method signature
    - formal parameters
      - number of arguments
      - Type
      - Order of arguments

Class and Object Examples

```
class bankAcc {

        /* instance vars */
        float balance;

        /* Constructor */
        bankAcc(float bal) {
                balance = bal;
        }

        /* methods */
        public void deposit(float d) {
                balance = balance+d;
        }

        public void withdraw(float d) {
                if (balance-d > 0){
                        balance = balance-d;
        }

        public void getBalance(){
                printf(balance);
        }
}
```

```
public int main() {

        samsAcc = new bankAcc(0);
        samsAcc.deposit(100);
        samsAcc.getBalance(); /*100*/

        samsAcc.withdraw(110);
        samsAcc.getBalance(); /*100*/

        samsAcc.withdraw(90);
        samsAcc.getBalance(); /*10*/

        return 1;
}
```

STANDARD LIBRARY FUNCTIONS

String manipulation

- Strlen
- Strcpy
- Strcmp
- Strcat
- Strstr

Mathematical functions for floating point integers

- Trigonometric function
- Exponent
- Logarithm
- Square root
- Random

TESTING

Testall.sh

- Microc’s test script
- Running tests ➔ Makefile overview
- Compares output with .out file

```
public class Student
{
   string uni;
   float grade;


   Student(string s)
   {
       uni = s;
       grade = 0.0;
   }


   string getUni()
   {
       return uni;
   }


   float getGrade()
   {
       return grade;
   }

   void addPoint(float p)
   {
       grade = grade + p;
   }


   void setGrade(float g)
   {
       grade = g;
   }
}
```

```
void main()
{
   Student s = new Student("gg123");

   int base = 100;
   int firstE = 0;
   int secondE = 0;
   int thirdE = 0;


   prints("What is square root of 256? (25
points) \n");
   prints("Student answer: 16 \n");
   if(sqrt(256) == 16)
   {
       s.addPoint(25.0);
       prints("Correct! Gained 25 points
\n");
   }
   else
   {
       prints("Wrong answer.\n");
       fristE = -25;
   }
```

```
prints("What is log of 4096? (35 points) \n");
 prints("Student answer: 12 \n");
 if(log(4096) == 12)
 {
     s.addPoint(35.0);
     prints("Correct! Gained 35 points");
 }
 else
 {
     prints("Wrong answer.\n");
     secondE = -35;
 }

 prints("What is sin of 0.7? (40 points) \n");
 prints("Student answer: 42 \n");
 if(sin(0.7) == 42)
 {
     s.addPoint(40.0);
     prints("Correct! Gained 40 points");
     thirdE = 40;
 }
 else
 {
     prints("Wrong answer.\n");
     thirdE = -40;
 }
```

```
    (*Drop the lowest exam")
   int lowestExam = Math.min(firstE,
Math.min(secondE, thirdE));
   base = base + lowestExam;
   float finalGrade = s.getGrade()/base;
   s.setGrade(finalGrade);
   prints(string_concat(s.getUni, "'s grade is: "));
   printf(s.getGrade);
}
```