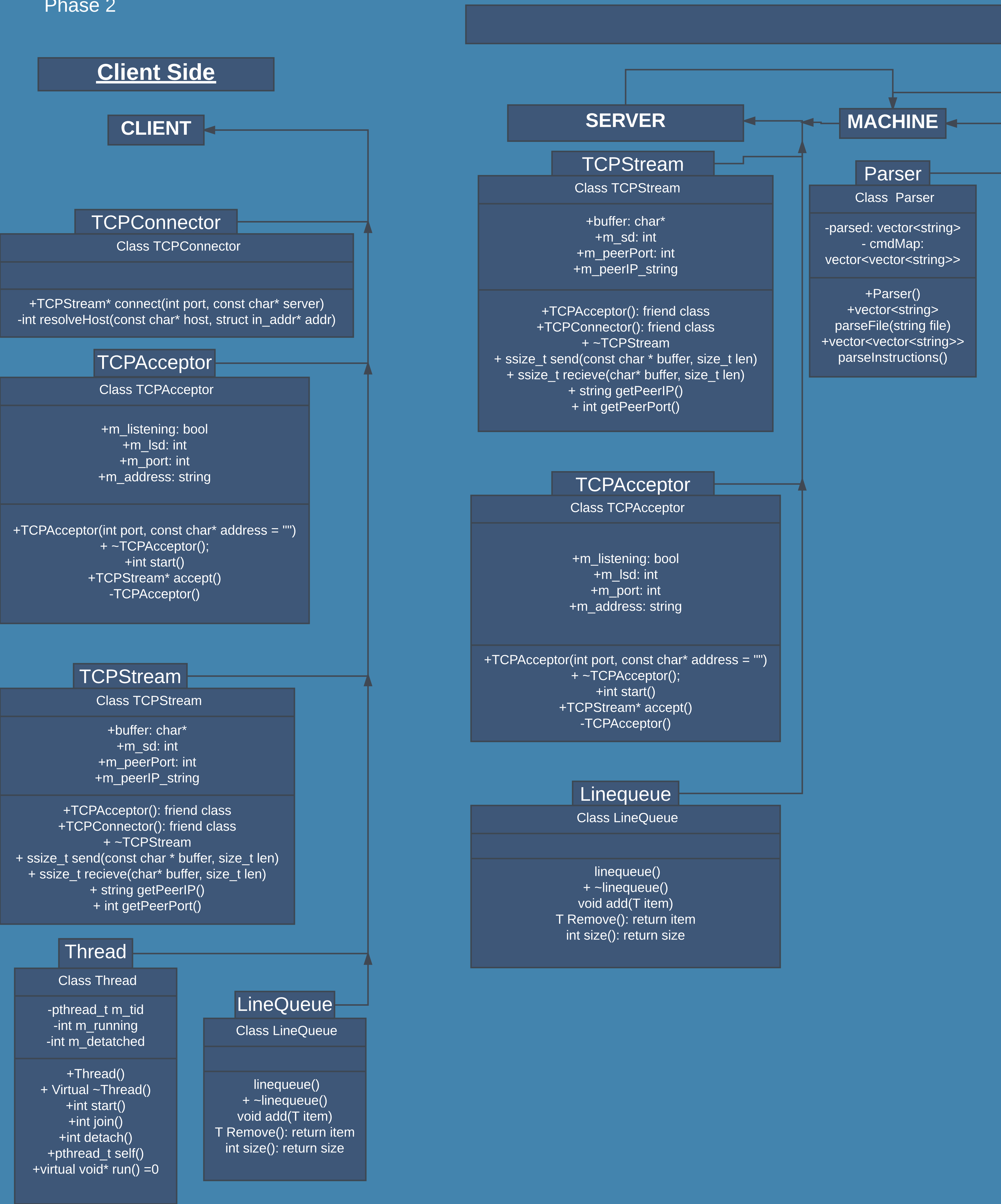
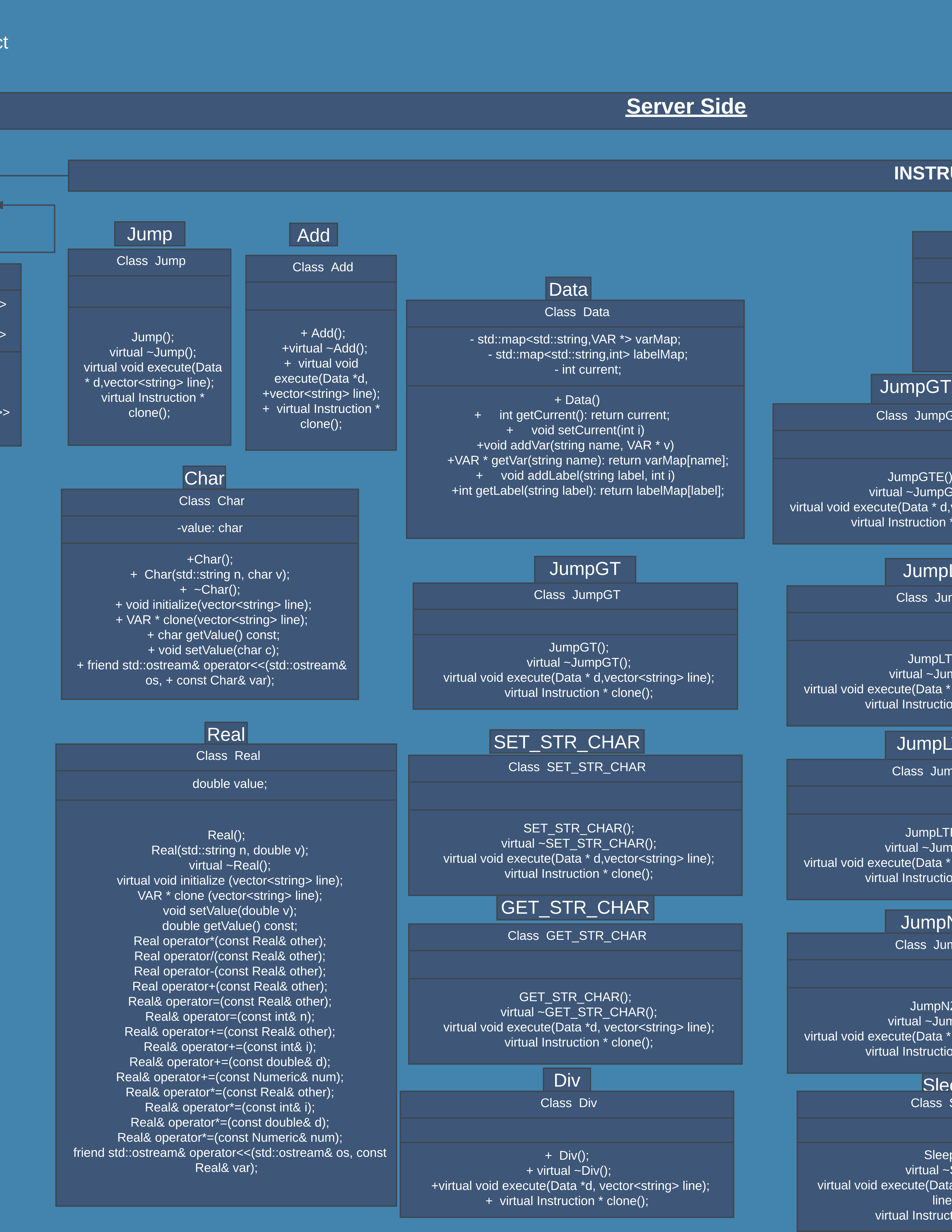
MIS Class Diagram

Phase 2

Brian Nguyen & Froy Valencia | CMPS109 MIS Project





Jump

Class Jump

```
Jump();
virtual ~Jump();
virtual void execute(Data
* d,vector<string> line);
virtual Instruction *
clone();
```

Add

Class Add

```
+ Add();
+virtual ~Add();
+ virtual void
execute(Data *d,
+vector<string> line);
+ virtual Instruction *
clone();
```

Data

Class Data

```
- std::map<std::string,VAR *> varMap;
- std::map<std::string,int> labelMap;
- int current;
```

```
+ Data()
+     int getCurrent(): return current;
+     void setCurrent(int i)
+void addVar(string name, VAR * v)
+VAR * getVar(string name): return varMap[name];
+     void addLabel(string label, int i)
+int getLabel(string label): return labelMap[label];
```

JumpGT

Class JumpG

```
JumpGTE()
virtual ~JumpG
virtual void execute(Data * d,v
virtual Instruction *
```

Char

Class Char

```
-value: char
```

```
+Char();
+ Char(std::string n, char v);
+ ~Char();
+ void initialize(vector<string> line);
+ VAR * clone(vector<string> line);
+ char getValue() const;
+ void setValue(char c);
+ friend std::ostream& operator<<(std::ostream&
os, + const Char& var);
```

JumpGT

Class JumpGT

```
JumpGT();
virtual ~JumpGT();
virtual void execute(Data * d,vector<string> line);
virtual Instruction * clone();
```

JumpL

Class Jur

```
JumpLT
virtual ~Jun
virtual void execute(Data *
virtual Instructio
```

Real

Class Real

```
double value;
```

```
Real();
Real(std::string n, double v);
virtual ~Real();
virtual void initialize (vector<string> line);
VAR * clone (vector<string> line);
void setValue(double v);
double getValue() const;
Real operator*(const Real& other);
Real operator/(const Real& other);
Real operator-(const Real& other);
Real operator+(const Real& other);
Real& operator=(const Real& other);
Real& operator=(const int& n);
Real& operator+=(const Real& other);
Real& operator+=(const int& i);
Real& operator+=(const double& d);
Real& operator+=(const Numeric& num);
Real& operator*=(const Real& other);
Real& operator*=(const int& i);
Real& operator*=(const double& d);
Real& operator*=(const Numeric& num);
friend std::ostream& operator<<(std::ostream& os, const
Real& var);
```

SET_STR_CHAR

Class SET_STR_CHAR

```
SET_STR_CHAR();
virtual ~SET_STR_CHAR();
virtual void execute(Data * d,vector<string> line);
virtual Instruction * clone();
```

JumpL

Class Jum

```
JumpLTI
virtual ~Jum
virtual void execute(Data *
virtual Instructio
```

GET_STR_CHAR

Class GET_STR_CHAR

```
GET_STR_CHAR();
virtual ~GET_STR_CHAR();
virtual void execute(Data *d, vector<string> line);
virtual Instruction * clone();
```

JumpN

Class Jun

```
JumpNZ
virtual ~Jun
virtual void execute(Data *
virtual Instructio
```

Div

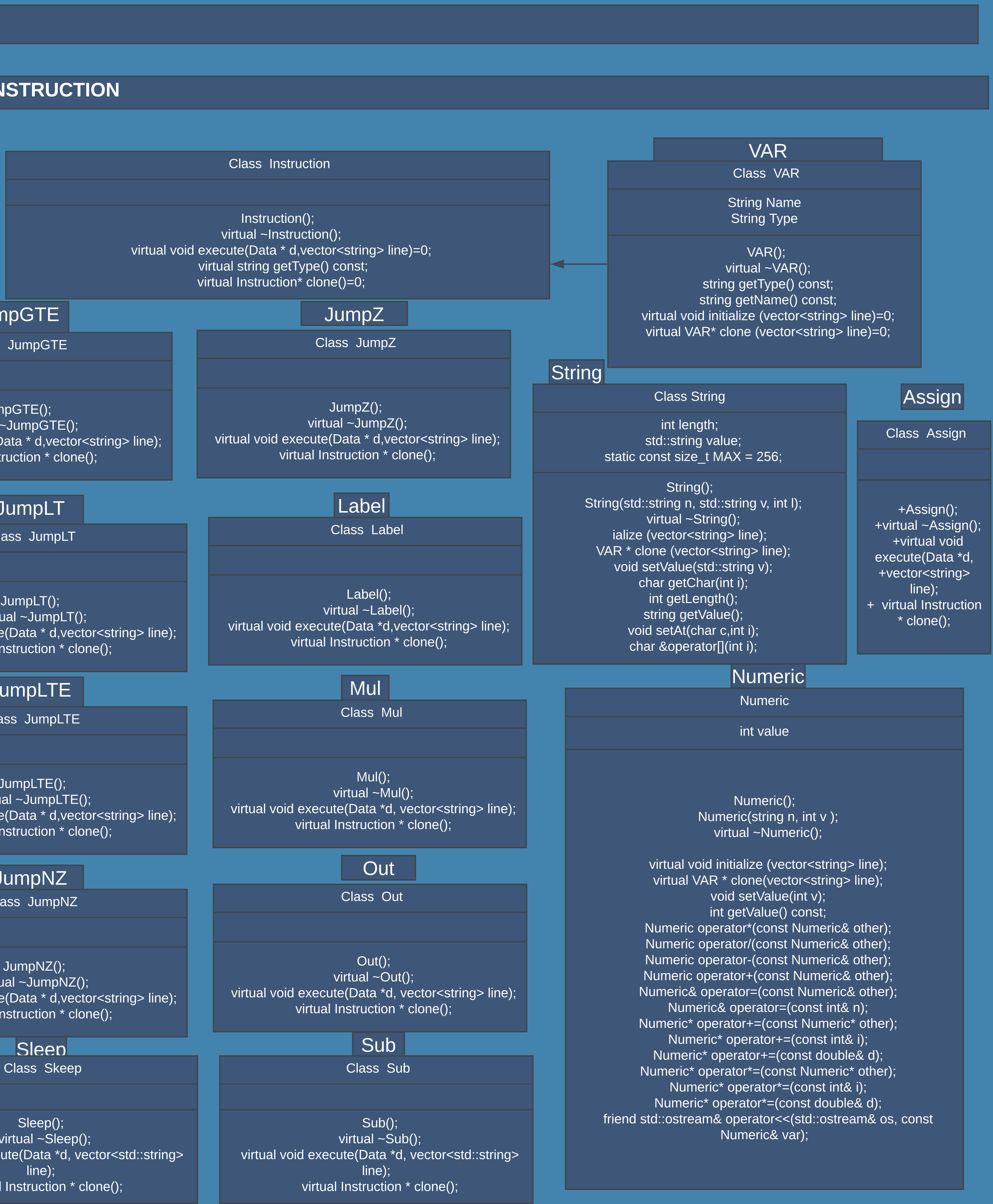
Class Div

```
+ Div();
+ virtual ~Div();
+virtual void execute(Data *d, vector<string> line);
+ virtual Instruction * clone();
```

Slee

Class S

```
Sleep
virtual ~S
virtual void execute(Data
line
virtual Instruct
```



NSTRUCTION

VAR

```
Class  VAR

String Name
String Type

VAR();
virtual ~VAR();
string getType() const;
string getName() const;
virtual void initialize (vector<string> line)=0;
virtual VAR* clone (vector<string> line)=0;
```

```
Class  Instruction

Instruction();
virtual ~Instruction();
virtual void execute(Data * d,vector<string> line)=0;
virtual string getType() const;
virtual Instruction* clone()=0;
```

mpGTE

```
 JumpGTE

npGTE();
~JumpGTE();
Data * d,vector<string> line);
truction * clone();
```

JumpZ

```
Class  JumpZ

JumpZ();
virtual ~JumpZ();
virtual void execute(Data * d,vector<string> line);
virtual Instruction * clone();
```

String

```
Class String

int length;
std::string value;
static const size_t MAX = 256;

String();
String(std::string n, std::string v, int l);
virtual ~String();
ialize (vector<string> line);
VAR * clone (vector<string> line);
void setValue(std::string v);
char getChar(int i);
int getLength();
string getValue();
void setAt(char c,int i);
char &operator[](int i);
```

Assign

```
Class  Assign

+Assign();
+virtual ~Assign();
+virtual void
execute(Data *d,
+vector<string>
line);
+  virtual Instruction
* clone();
```

JumpLT

```
ass  JumpLT

JumpLT();
ual ~JumpLT();
e(Data * d,vector<string> line);
nstruction * clone();
```

Label

```
Class  Label

Label();
virtual ~Label();
virtual void execute(Data *d,vector<string> line);
virtual Instruction * clone();
```

umpLTE

```
ass  JumpLTE

JumpLTE();
al ~JumpLTE();
e(Data * d,vector<string> line);
nstruction * clone();
```

Mul

```
Class  Mul

Mul();
virtual ~Mul();
virtual void execute(Data *d, vector<string> line);
virtual Instruction * clone();
```

Numeric

```
Numeric

int value

Numeric();
Numeric(string n, int v );
virtual ~Numeric();

virtual void initialize (vector<string> line);
virtual VAR * clone(vector<string> line);
void setValue(int v);
int getValue() const;
Numeric operator*(const Numeric& other);
Numeric operator/(const Numeric& other);
Numeric operator-(const Numeric& other);
Numeric operator+(const Numeric& other);
Numeric& operator=(const Numeric& other);
Numeric& operator=(const int& n);
Numeric* operator+=(const Numeric* other);
Numeric* operator+=(const int& i);
Numeric* operator+=(const double& d);
Numeric* operator*=(const Numeric* other);
Numeric* operator*=(const int& i);
Numeric* operator*=(const double& d);
friend std::ostream& operator<<(std::ostream& os, const
Numeric& var);
```

JumpNZ

```
ass  JumpNZ

 JumpNZ();
ual ~JumpNZ();
e(Data * d,vector<string> line);
nstruction * clone();
```

Out

```
Class  Out

Out();
virtual ~Out();
virtual void execute(Data *d, vector<string> line);
virtual Instruction * clone();
```

Sleep

```
 Class  Skeep

Sleep();
virtual ~Sleep();
ute(Data *d, vector<std::string>
line);
l Instruction * clone();
```

Sub

```
Class  Sub

Sub();
virtual ~Sub();
virtual void execute(Data *d, vector<std::string>
line);
virtual Instruction * clone();
```