

# Compiling to WebAssembly

Seminar project by Vincent Hofer

Assisted by Manuel Leuenberger & Oli Flückiger

# Project goals

- Explore parser generators and WebAssembly specification
- Build a compiler pipeline targeting WebAssembly
- Develop a new simple language without prior compiling knowledge

# What is WebAssembly?

“WebAssembly is a new type of code that can be run in modern web browsers. It is a binary instruction format for a stack-based virtual machine”

- Near-native performance
- Supported by all main browser manufacturers
- Has a textual representation, besides binary format
- Intended as compilation target for languages like C/C++



# WebAssembly Text Format

```
def main {
    var x;
    x = 5;
    call foo(x);
};

def foo(num) {
    num*2;
};

(module
    (func $main (result f64)
        (local $x f64)
        (set_local $x (f64.const 5))
        (call $foo (get_local $x))
    )
    (func $foo (param $num f64) (result f64)
        (f64.mul (get_local $num)(f64.const 2))
    )
    (export "main" (func $main))
)
```

# Development of my language and compiler

Iteration 1

$2*3+(1+2)/3$

# Development of my language and compiler

## Iteration 2

```
var x;  
x = 3;  
x + 2;
```

# Development of my language and compiler

## Iteration 3

```
def main{
    var a;
    a = 3;
    call add(a)(2);
};
```

```
def add(x)(y){
    x+y;
};
```

# Development of my language and compiler

## Iteration 4

```
def fib(x) {  
    if (x<=2) {  
        x=1;  
    } else {  
        x = call fib(x-2) + call fib(x-1);  
    };  
    x;  
};
```

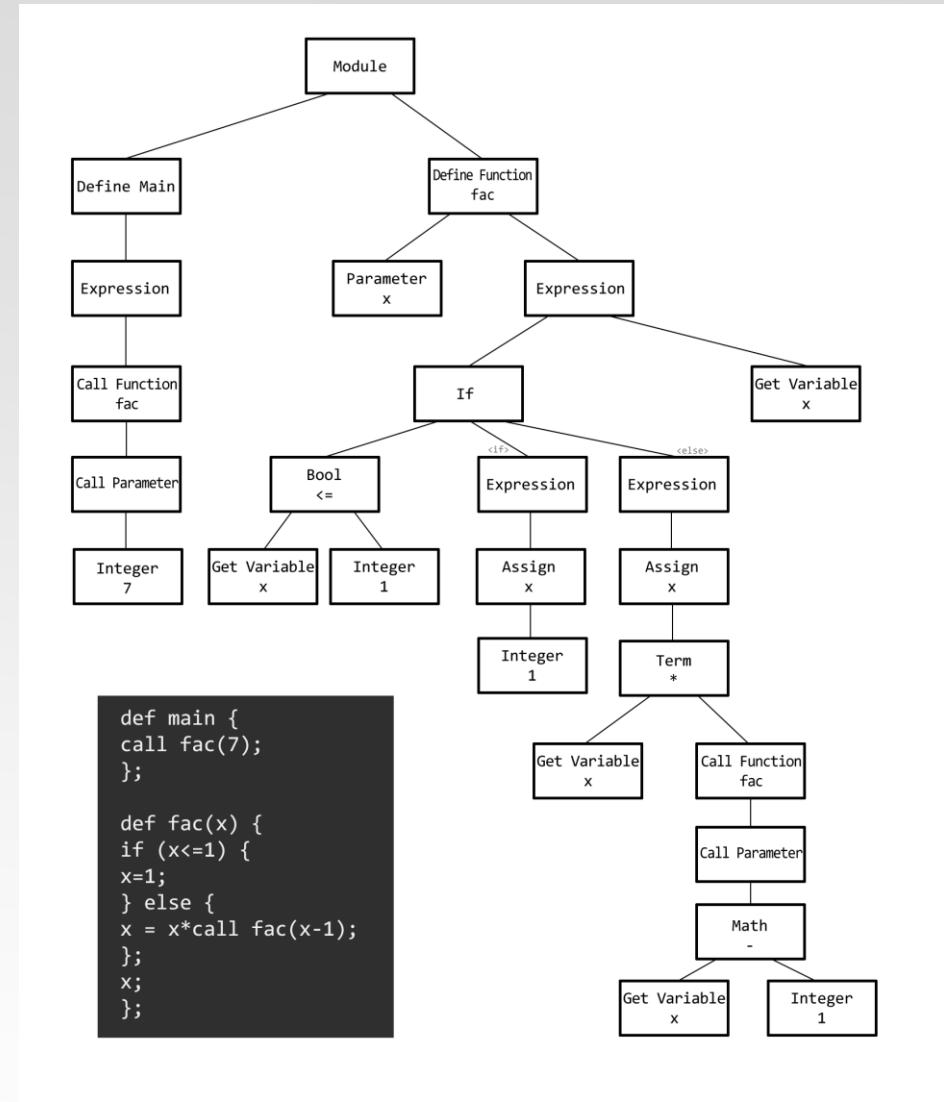
# Development of my language and compiler

## Iteration 5

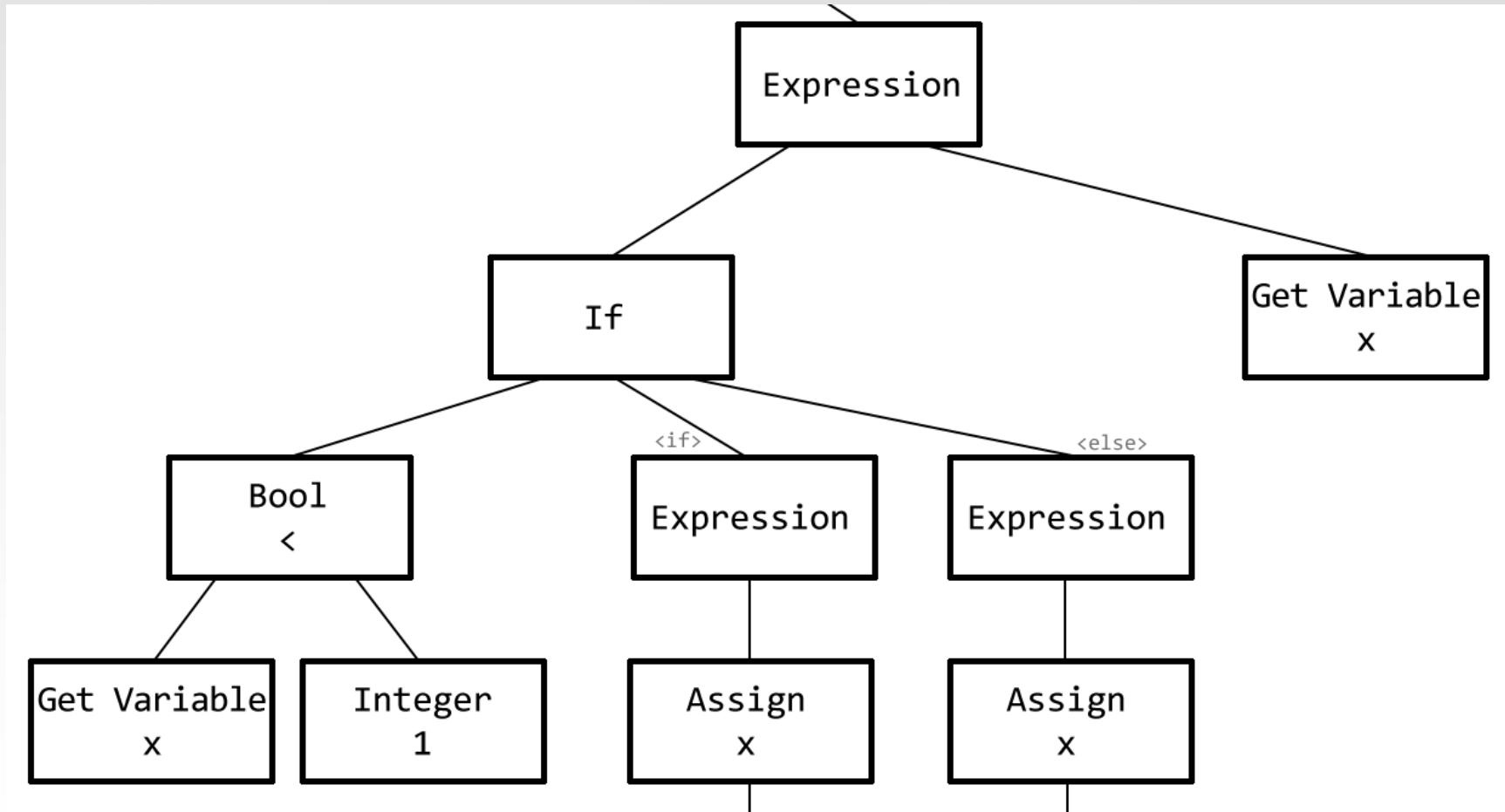
```
def main {  
    var x;  
    x = 3;  
    array a [1 2 x 4 5];  
    len a + get a [2];  
};
```

# Parser and AST

- Parser generator: PEG.js
- In the beginning: Direct output of text format
- Later: Secondary step through AST



# Parser and AST



# Squareroot implementation

*WebAssembly Compiler Suite by Manuel Leuenberger*

Source	load	save	Grammar	load	save		A	C	W	Output	load	save
1 def main { 2     sqrt(2); 3 };	144         accept() { 145             return visit.visitTerm(this); 146         } 147     } 148 149     class Parameter extends Node { 150         constructor(char) { 151             super(); 152             this.name = "Parameter"; 153             this.data = [char]; 154         } 155         accept() { 156             return visit.visitParameter(this); 157         } 158     } 159 160     class Squareroot extends Node { 161         constructor(math) { 162             super(); 163             this.name = "Squareroot"; 164             this.children = [math]; 165         } 166         accept() { 167             return visit.visitSquareroot(this); 168         } 169     } 170 171     class Factor extends Node { 172         constructor(exp) { 173             super(); 174             this.name = "Factor"; 175             this.children = [exp]; 176         } 177         accept() { 178             return visit.visitFactor(this); 179         } 180     } 181 182     class CreateArray extends Node { 183         constructor(char,elements) { 184             super(); 185             this.name = "Array"; 186             this.data = [char]; 187         } 188         accept() { 189             return visit.visitCreateArray(this); 190         } 191     } 192 193     class Identifier extends Node { 194         constructor(name) { 195             super(); 196             this.name = name; 197             this.data = [name]; 198         } 199         accept() { 200             return visit.visitIdentifier(this); 201         } 202     } 203 204     class Constant extends Node { 205         constructor(data) { 206             super(); 207             this.name = "Constant"; 208             this.data = [data]; 209         } 210         accept() { 211             return visit.visitConstant(this); 212         } 213     } 214 215     class Operator extends Node { 216         constructor(operator) { 217             super(); 218             this.name = operator; 219             this.data = [operator]; 220         } 221         accept() { 222             return visit.visitOperator(this); 223         } 224     } 225 226     class LeftBrace extends Node { 227         constructor() { 228             super(); 229             this.name = "LeftBrace"; 230         } 231         accept() { 232             return visit.visitLeftBrace(this); 233         } 234     } 235 236     class RightBrace extends Node { 237         constructor() { 238             super(); 239             this.name = "RightBrace"; 240         } 241         accept() { 242             return visit.visitRightBrace(this); 243         } 244     } 245 246     class LeftParanthesis extends Node { 247         constructor() { 248             super(); 249             this.name = "LeftParanthesis"; 250         } 251         accept() { 252             return visit.visitLeftParanthesis(this); 253         } 254     } 255 256     class RightParanthesis extends Node { 257         constructor() { 258             super(); 259             this.name = "RightParanthesis"; 260         } 261         accept() { 262             return visit.visitRightParanthesis(this); 263         } 264     } 265 266     class LeftBracket extends Node { 267         constructor() { 268             super(); 269             this.name = "LeftBracket"; 270         } 271         accept() { 272             return visit.visitLeftBracket(this); 273         } 274     } 275 276     class RightBracket extends Node { 277         constructor() { 278             super(); 279             this.name = "RightBracket"; 280         } 281         accept() { 282             return visit.visitRightBracket(this); 283         } 284     } 285 286     class Dot extends Node { 287         constructor() { 288             super(); 289             this.name = "Dot"; 290         } 291         accept() { 292             return visit.visitDot(this); 293         } 294     } 295 296     class Comma extends Node { 297         constructor() { 298             super(); 299             this.name = "Comma"; 300         } 301         accept() { 302             return visit.visitComma(this); 303         } 304     } 305 306     class Semicolon extends Node { 307         constructor() { 308             super(); 309             this.name = "Semicolon"; 310         } 311         accept() { 312             return visit.visitSemicolon(this); 313         } 314     } 315 316     class Colon extends Node { 317         constructor() { 318             super(); 319             this.name = "Colon"; 320         } 321         accept() { 322             return visit.visitColon(this); 323         } 324     } 325 326     class Plus extends Node { 327         constructor() { 328             super(); 329             this.name = "Plus"; 330         } 331         accept() { 332             return visit.visitPlus(this); 333         } 334     } 335 336     class Minus extends Node { 337         constructor() { 338             super(); 339             this.name = "Minus"; 340         } 341         accept() { 342             return visit.visitMinus(this); 343         } 344     } 345 346     class Multiplie extends Node { 347         constructor() { 348             super(); 349             this.name = "Multiplie"; 350         } 351         accept() { 352             return visit.visitMultiplie(this); 353         } 354     } 355 356     class Divide extends Node { 357         constructor() { 358             super(); 359             this.name = "Divide"; 360         } 361         accept() { 362             return visit.visitDivide(this); 363         } 364     } 365 366     class Modulus extends Node { 367         constructor() { 368             super(); 369             this.name = "Modulus"; 370         } 371         accept() { 372             return visit.visitModulus(this); 373         } 374     } 375 376     class LessThan extends Node { 377         constructor() { 378             super(); 379             this.name = "LessThan"; 380         } 381         accept() { 382             return visit.visitLessThan(this); 383         } 384     } 385 386     class GreaterThan extends Node { 387         constructor() { 388             super(); 389             this.name = "GreaterThan"; 390         } 391         accept() { 392             return visit.visitGreaterThan(this); 393         } 394     } 395 396     class LessOrEqual extends Node { 397         constructor() { 398             super(); 399             this.name = "LessOrEqual"; 400         } 401         accept() { 402             return visit.visitLessOrEqual(this); 403         } 404     } 405 406     class GreaterOrEqual extends Node { 407         constructor() { 408             super(); 409             this.name = "GreaterOrEqual"; 410         } 411         accept() { 412             return visit.visitGreaterOrEqual(this); 413         } 414     } 415 416     class Equal extends Node { 417         constructor() { 418             super(); 419             this.name = "Equal"; 420         } 421         accept() { 422             return visit.visitEqual(this); 423         } 424     } 425 426     class NotEqual extends Node { 427         constructor() { 428             super(); 429             this.name = "NotEqual"; 430         } 431         accept() { 432             return visit.visitNotEqual(this); 433         } 434     } 435 436     class And extends Node { 437         constructor() { 438             super(); 439             this.name = "And"; 440         } 441         accept() { 442             return visit.visitAnd(this); 443         } 444     } 445 446     class Or extends Node { 447         constructor() { 448             super(); 449             this.name = "Or"; 450         } 451         accept() { 452             return visit.visitOr(this); 453         } 454     } 455 456     class Not extends Node { 457         constructor() { 458             super(); 459             this.name = "Not"; 460         } 461         accept() { 462             return visit.visitNot(this); 463         } 464     } 465 466     class LeftSquareBrace extends Node { 467         constructor() { 468             super(); 469             this.name = "LeftSquareBrace"; 470         } 471         accept() { 472             return visit.visitLeftSquareBrace(this); 473         } 474     } 475 476     class RightSquareBrace extends Node { 477         constructor() { 478             super(); 479             this.name = "RightSquareBrace"; 480         } 481         accept() { 482             return visit.visitRightSquareBrace(this); 483         } 484     } 485 486     class LeftAngleBrace extends Node { 487         constructor() { 488             super(); 489             this.name = "LeftAngleBrace"; 490         } 491         accept() { 492             return visit.visitLeftAngleBrace(this); 493         } 494     } 495 496     class RightAngleBrace extends Node { 497         constructor() { 498             super(); 499             this.name = "RightAngleBrace"; 500         } 501         accept() { 502             return visit.visitRightAngleBrace(this); 503         } 504     } 505 506     class LeftCurlyBrace extends Node { 507         constructor() { 508             super(); 509             this.name = "LeftCurlyBrace"; 510         } 511         accept() { 512             return visit.visitLeftCurlyBrace(this); 513         } 514     } 515 516     class RightCurlyBrace extends Node { 517         constructor() { 518             super(); 519             this.name = "RightCurlyBrace"; 520         } 521         accept() { 522             return visit.visitRightCurlyBrace(this); 523         } 524     } 525 526     class LeftAngleBrace2 extends Node { 527         constructor() { 528             super(); 529             this.name = "LeftAngleBrace2"; 530         } 531         accept() { 532             return visit.visitLeftAngleBrace2(this); 533         } 534     } 535 536     class RightAngleBrace2 extends Node { 537         constructor() { 538             super(); 539             this.name = "RightAngleBrace2"; 540         } 541         accept() { 542             return visit.visitRightAngleBrace2(this); 543         } 544     } 545 546     class LeftCurlyBrace2 extends Node { 547         constructor() { 548             super(); 549             this.name = "LeftCurlyBrace2"; 550         } 551         accept() { 552             return visit.visitLeftCurlyBrace2(this); 553         } 554     } 555 556     class RightCurlyBrace2 extends Node { 557         constructor() { 558             super(); 559             this.name = "RightCurlyBrace2"; 560         } 561         accept() { 562             return visit.visitRightCurlyBrace2(this); 563         } 564     } 565 566     class LeftAngleBrace3 extends Node { 567         constructor() { 568             super(); 569             this.name = "LeftAngleBrace3"; 570         } 571         accept() { 572             return visit.visitLeftAngleBrace3(this); 573         } 574     } 575 576     class RightAngleBrace3 extends Node { 577         constructor() { 578             super(); 579             this.name = "RightAngleBrace3"; 580         } 581         accept() { 582             return visit.visitRightAngleBrace3(this); 583         } 584     } 585 586     class LeftCurlyBrace3 extends Node { 587         constructor() { 588             super(); 589             this.name = "LeftCurlyBrace3"; 590         } 591         accept() { 592             return visit.visitLeftCurlyBrace3(this); 593         } 594     } 595 596     class RightCurlyBrace3 extends Node { 597         constructor() { 598             super(); 599             this.name = "RightCurlyBrace3"; 600         } 601         accept() { 602             return visit.visitRightCurlyBrace3(this); 603         } 604     } 605 606     class LeftAngleBrace4 extends Node { 607         constructor() { 608             super(); 609             this.name = "LeftAngleBrace4"; 610         } 611         accept() { 612             return visit.visitLeftAngleBrace4(this); 613         } 614     } 615 616     class RightAngleBrace4 extends Node { 617         constructor() { 618             super(); 619             this.name = "RightAngleBrace4"; 620         } 621         accept() { 622             return visit.visitRightAngleBrace4(this); 623         } 624     } 625 626     class LeftCurlyBrace4 extends Node { 627         constructor() { 628             super(); 629             this.name = "LeftCurlyBrace4"; 630         } 631         accept() { 632             return visit.visitLeftCurlyBrace4(this); 633         } 634     } 635 636     class RightCurlyBrace4 extends Node { 637         constructor() { 638             super(); 639             this.name = "RightCurlyBrace4"; 640         } 641         accept() { 642             return visit.visitRightCurlyBrace4(this); 643         } 644     } 645 646     class LeftAngleBrace5 extends Node { 647         constructor() { 648             super(); 649             this.name = "LeftAngleBrace5"; 650         } 651         accept() { 652             return visit.visitLeftAngleBrace5(this); 653         } 654     } 655 656     class RightAngleBrace5 extends Node { 657         constructor() { 658             super(); 659             this.name = "RightAngleBrace5"; 660         } 661         accept() { 662             return visit.visitRightAngleBrace5(this); 663         } 664     } 665 666     class LeftCurlyBrace5 extends Node { 667         constructor() { 668             super(); 669             this.name = "LeftCurlyBrace5"; 670         } 671         accept() { 672             return visit.visitLeftCurlyBrace5(this); 673         } 674     } 675 676     class RightCurlyBrace5 extends Node { 677         constructor() { 678             super(); 679             this.name = "RightCurlyBrace5"; 680         } 681         accept() { 682             return visit.visitRightCurlyBrace5(this); 683         } 684     } 685 686     class LeftAngleBrace6 extends Node { 687         constructor() { 688             super(); 689             this.name = "LeftAngleBrace6"; 690         } 691         accept() { 692             return visit.visitLeftAngleBrace6(this); 693         } 694     } 695 696     class RightAngleBrace6 extends Node { 697         constructor() { 698             super(); 699             this.name = "RightAngleBrace6"; 700         } 701         accept() { 702             return visit.visitRightAngleBrace6(this); 703         } 704     } 705 706     class LeftCurlyBrace6 extends Node { 707         constructor() { 708             super(); 709             this.name = "LeftCurlyBrace6"; 710         } 711         accept() { 712             return visit.visitLeftCurlyBrace6(this); 713         } 714     } 715 716     class RightCurlyBrace6 extends Node { 717         constructor() { 718             super(); 719             this.name = "RightCurlyBrace6"; 720         } 721         accept() { 722             return visit.visitRightCurlyBrace6(this); 723         } 724     } 725 726     class LeftAngleBrace7 extends Node { 727         constructor() { 728             super(); 729             this.name = "LeftAngleBrace7"; 730         } 731         accept() { 732             return visit.visitLeftAngleBrace7(this); 733         } 734     } 735 736     class RightAngleBrace7 extends Node { 737         constructor() { 738             super(); 739             this.name = "RightAngleBrace7"; 740         } 741         accept() { 742             return visit.visitRightAngleBrace7(this); 743         } 744     } 745 746     class LeftCurlyBrace7 extends Node { 747         constructor() { 748             super(); 749             this.name = "LeftCurlyBrace7"; 750         } 751         accept() { 752             return visit.visitLeftCurlyBrace7(this); 753         } 754     } 755 756     class RightCurlyBrace7 extends Node { 757         constructor() { 758             super(); 759             this.name = "RightCurlyBrace7"; 760         } 761         accept() { 762             return visit.visitRightCurlyBrace7(this); 763         } 764     } 765 766     class LeftAngleBrace8 extends Node { 767         constructor() { 768             super(); 769             this.name = "LeftAngleBrace8"; 770         } 771         accept() { 772             return visit.visitLeftAngleBrace8(this); 773         } 774     } 775 776     class RightAngleBrace8 extends Node { 777         constructor() { 778             super(); 779             this.name = "RightAngleBrace8"; 780         } 781         accept() { 782             return visit.visitRightAngleBrace8(this); 783         } 784     } 785 786     class LeftCurlyBrace8 extends Node { 787         constructor() { 788             super(); 789             this.name = "LeftCurlyBrace8"; 790         } 791         accept() { 792             return visit.visitLeftCurlyBrace8(this); 793         } 794     } 795 796     class RightCurlyBrace8 extends Node { 797         constructor() { 798             super(); 799             this.name = "RightCurlyBrace8"; 800         } 801         accept() { 802             return visit.visitRightCurlyBrace8(this); 803         } 804     } 805 806     class LeftAngleBrace9 extends Node { 807         constructor() { 808             super(); 809             this.name = "LeftAngleBrace9"; 810         } 811         accept() { 812             return visit.visitLeftAngleBrace9(this); 813         } 814     } 815 816     class RightAngleBrace9 extends Node { 817         constructor() { 818             super(); 819             this.name = "RightAngleBrace9"; 820         } 821         accept() { 822             return visit.visitRightAngleBrace9(this); 823         } 824     } 825 826     class LeftCurlyBrace9 extends Node { 827         constructor() { 828             super(); 829             this.name = "LeftCurlyBrace9"; 830         } 831         accept() { 832             return visit.visitLeftCurlyBrace9(this); 833         } 834     } 835 836     class RightCurlyBrace9 extends Node { 837         constructor() { 838             super(); 839             this.name = "RightCurlyBrace9"; 840         } 841         accept() { 842             return visit.visitRightCurlyBrace9(this); 843         } 844     } 845 846     class LeftAngleBrace10 extends Node { 847         constructor() { 848             super(); 849             this.name = "LeftAngleBrace10"; 850         } 851         accept() { 852             return visit.visitLeftAngleBrace10(this); 853         } 854     } 855 856     class RightAngleBrace10 extends Node { 857         constructor() { 858             super(); 859             this.name = "RightAngleBrace10"; 860         } 861         accept() { 862             return visit.visitRightAngleBrace10(this); 863         } 864     } 865 866     class LeftCurlyBrace10 extends Node { 867         constructor() { 868             super(); 869             this.name = "LeftCurlyBrace10"; 870         } 871         accept() { 872             return visit.visitLeftCurlyBrace10(this); 873         } 874     } 875 876     class RightCurlyBrace10 extends Node { 877         constructor() { 878             super(); 879             this.name = "RightCurlyBrace10"; 880         } 881         accept() { 882             return visit.visitRightCurlyBrace10(this); 883         } 884     } 885 886     class LeftAngleBrace11 extends Node { 887         constructor() { 888             super(); 889             this.name = "LeftAngleBrace11"; 890         } 891         accept() { 892             return visit.visitLeftAngleBrace11(this); 893         } 894     } 895 896     class RightAngleBrace11 extends Node { 897         constructor() { 898             super(); 899             this.name = "RightAngleBrace11"; 900         } 901         accept() { 902             return visit.visitRightAngleBrace11(this); 903         } 904     } 905 906     class LeftCurlyBrace11 extends Node { 907         constructor() { 908             super(); 909             this.name = "LeftCurlyBrace11"; 910         } 911         accept() { 912             return visit.visitLeftCurlyBrace11(this); 913         } 914     } 915 916     class RightCurlyBrace11 extends Node { 917         constructor() { 918             super(); 919             this.name = "RightCurlyBrace11"; 920         } 921         accept() { 922             return visit.visitRightCurlyBrace11(this); 923         } 924     } 925 926     class LeftAngleBrace12 extends Node { 927         constructor() { 928             super(); 929             this.name = "LeftAngleBrace12"; 930         } 931         accept() { 932             return visit.visitLeftAngleBrace12(this); 933         } 934     } 935 936     class RightAngleBrace12 extends Node { 937         constructor() { 938             super(); 939             this.name = "RightAngleBrace12"; 940         } 941         accept() { 942             return visit.visitRightAngleBrace12(this); 943         } 944     } 945 946     class LeftCurlyBrace12 extends Node { 947         constructor() { 948             super(); 949             this.name = "LeftCurlyBrace12"; 950         } 951         accept() { 952             return visit.visitLeftCurlyBrace12(this); 953         } 954     } 955 956     class RightCurlyBrace12 extends Node { 957         constructor() { 958             super(); 959             this.name = "RightCurlyBrace12"; 960         } 961         accept() { 962             return visit.visitRightCurlyBrace12(this); 963         } 964     } 965 966     class LeftAngleBrace13 extends Node { 967         constructor() { 968             super(); 969             this.name = "LeftAngleBrace13"; 970         } 971         accept() { 972             return visit.visitLeftAngleBrace13(this); 973         } 974     } 975 976     class RightAngleBrace13 extends Node { 977         constructor() { 978             super(); 979             this.name = "RightAngleBrace13"; 980         } 981         accept() { 982             return visit.visitRightAngleBrace13(this); 983         } 984     } 985 986     class LeftCurlyBrace13 extends Node { 987         constructor() { 988             super(); 989             this.name = "LeftCurlyBrace13"; 990         } 991         accept() { 992             return visit.visitLeftCurlyBrace13(this); 993         } 994     } 995 996     class RightCurlyBrace13 extends Node { 997         constructor() { 998             super(); 999             this.name = "RightCurlyBrace13"; 1000        } 1001        accept() { 1002            return visit.visitRightCurlyBrace13(this); 1003        } 1004    } 1005 1006    class Identifier extends Node { 1007        constructor(name) { 1008            super(); 1009            this.name = name; 1010        } 1011        accept() { 1012            return visit.visitIdentifier(this); 1013        } 1014    } 1015 1016    class Constant extends Node { 1017        constructor(data) { 1018            super(); 1019            this.name = "Constant"; 1020            this.data = [data]; 1021        } 1022        accept() { 1023            return visit.visitConstant(this); 1024        } 1025    } 1026 1027    class Operator extends Node { 1028        constructor(operator) { 1029            super(); 1030            this.name = operator; 1031            this.data = [operator]; 1032        } 1033        accept() { 1034            return visit.visitOperator(this); 1035        } 1036    } 1037 1038    class LeftParanthesis extends Node { 1039        constructor() { 1040            super(); 1041            this.name = "LeftParanthesis"; 1042        } 1043        accept() { 1044            return visit.visitLeftParanthesis(this); 1045        } 1046    } 1047 1048    class RightParanthesis extends Node { 1049        constructor() { 1050            super(); 1051            this.name = "RightParanthesis"; 1052        } 1053        accept() { 1054            return visit.visitRightParanthesis(this); 1055        } 1056    } 1057 1058    class LeftSquareBrace extends Node { 1059        constructor() { 1060            super(); 1061            this.name = "LeftSquareBrace"; 1062        } 1063        accept() { 1064            return visit.visitLeftSquareBrace(this); 1065        } 1066    } 1067 1068    class RightSquareBrace extends Node { 1069        constructor() { 1070            super(); 1071            this.name = "RightSquareBrace"; 1072        } 1073        accept() { 1074            return visit.visitRightSquareBrace(this); 1075        } 1076    } 1077 1078    class LeftAngleBrace extends Node { 1079        constructor() { 1080            super(); 1081            this.name = "LeftAngleBrace"; 1082        } 1083        accept() { 1084            return visit.visitLeftAngleBrace(this); 1085        } 1086    } 1087 1088    class RightAngleBrace extends Node { 1089        constructor() { 1090            super(); 1091            this.name = "RightAngleBrace"; 1092        } 1093        accept() { 1094            return visit.visitRightAngleBrace(this); 1095        } 1096    } 1097 1098    class LeftCurlyBrace extends Node { 1099        constructor() { 1100            super(); 1101            this.name = "LeftCurlyBrace"; 1102        } 1103        accept() { 1104            return visit.visitLeftCurlyBrace(this); 1105        } 1106    } 1107 1108    class RightCurlyBrace extends Node { 1109        constructor() { 1110            super(); 1111            this.name = "RightCurlyBrace"; 1112        } 1113        accept() { 1114            return visit.visitRightCurlyBrace(this); 1115        } 1116    } 1117 1118    class LeftAngleBrace2 extends Node { 1119        constructor() { 1120            super(); 1121            this.name = "LeftAngleBrace2"; 1122        } 1123        accept() { 1124            return visit.visitLeftAngleBrace2(this); 1125        } 1126    } 1127 1128    class RightAngleBrace2 extends Node { 1129        constructor() { 1130            super(); 1131            this.name = "RightAngleBrace2"; 1132        } 1133        accept() { 1134            return visit.visitRightAngleBrace2(this); 1135        } 1136    } 1137 1138    class LeftCurlyBrace2 extends Node { 1139        constructor() { 1140            super(); 1141            this.name = "LeftCurlyBrace2"; 1142        } 1143        accept() { 1144            return visit.visitLeftCurlyBrace2(this); 1145        } 1146    } 1147 1148    class RightCurlyBrace2											



S	G	AST	load	save	C	WAT	load	save	O
		<pre> 1 { 2   "name": "Module", 3   "data": [], 4   "children": [ 5     { 6       "name": "Define Main", 7       "data": [], 8       "children": [ 9         { 10          "name": "Expression", 11          "data": [], 12          "children": [ 13            [ 14              { 15                "name": "Squareroot", 16                "data": [], 17                "children": [ 18                  { 19                    "name": "Integer", 20                    "data": [ 21                      "2" 22                    ], 23                    "children": [] 24                  } 25                ] 26              }, 27              null 28            ] 29          ] 30        } 31      ] 32    }, 33    [] 34  ] 35 }</pre>			<pre> 1 (module 2 (memory \$0 1) 3 (func \$main (result f64) 4 (f64.sqrt (f64.const 2)) 5 ) 6 (export "main" (func \$main)) 7 )</pre>				

# WebAssembly vs. JavaScript

```
def fib(x) {  
    if (x<=2) {  
        x=1;  
    } else {  
        x = call fib(x-2) +  
            call fib(x-1);  
    };  
    x;  
};
```

```
function fib(x) {  
    if (x <= 2) return 1;  
    return fib(x-2) +  
        fib(x-1);  
}
```

# Summary

- WebAssembly can be used alongside JavaScript
- Learned about compilers, grammars and ASTs
- Wrote a language targeting WebAssembly
- Compiler can be expanded in the future, e.g.
  - » Data types
  - » Strings
  - » Objects