*whitespace* = *whitespaceitem* *whitespace?* . *whitespaceitem* = *linebreak* .  
*whitespaceitem* = *inlinespace* .  
*whitespaceitem* = *comment* .  
*whitespaceitem* = *multilinecomment* .  
*whitespaceitem* = U+0000, U+000B, or U+000C  
  
*linebreak* = U+000A  
*linebreak* = U+000D  
*linebreak* = U+000D followed by U+000A  
  
*inlinespaces* = *inlinespace* *inlinespaces?* .  
*inlinespace* = U+0009 or U+0020  
  
*comment* = **”//“** *commenttext* *linebreak* .  
*multilinecomment* = **”/\*“** *multilinecommenttext* **”\*/“** .  
  
*commenttext* = *commenttextitem* *commenttext?* .  
*commenttextitem* = Any Unicode scalar value except U+000A or U+000D  
  
*multilinecommenttext* = *multilinecommenttextitem* *multilinecommenttext?* .  
*multilinecommenttextitem* = *multilinecomment* .  
*multilinecommenttextitem* = *commenttextitem* .  
*multilinecommenttextitem* = Any Unicode scalar value except **”/\*“** or **”\*/“**  
  
*identifier* = *identifierhead* *identifiercharacters?* .  
*identifier* = **”`“** *identifierhead* *identifiercharacters?* **”`“** .  
*identifier* = *implicitparametername* .  
*identifier* = *propertywrapperprojection* .  
*identifierlist* = *identifier* | *identifier* **”,“** *identifierlist* .  
  
*identifierhead* = Upper or lowercase letter A through Z  
*identifierhead* = **”\_“** .  
*identifierhead* = U+00A8, U+00AA, U+00AD, U+00AF, U+00B2–U+00B5, or U+00B7–U+00BA  
*identifierhead* = U+00BC–U+00BE, U+00C0–U+00D6, U+00D8–U+00F6, or U+00F8–U+00FF  
*identifierhead* = U+0100–U+02FF, U+0370–U+167F, U+1681–U+180D, or U+180F–U+1DBF  
*identifierhead* = U+1E00–U+1FFF  
*identifierhead* = U+200B–U+200D, U+202A–U+202E, U+203F–U+2040, U+2054, or U+2060–U+206F  
*identifierhead* = U+2070–U+20CF, U+2100–U+218F, U+2460–U+24FF, or U+2776–U+2793  
*identifierhead* = U+2C00–U+2DFF or U+2E80–U+2FFF  
*identifierhead* = U+3004–U+3007, U+3021–U+302F, U+3031–U+303F, or U+3040–U+D7FF  
*identifierhead* = U+F900–U+FD3D, U+FD40–U+FDCF, U+FDF0–U+FE1F, or U+FE30–U+FE44  
*identifierhead* = U+FE47–U+FFFD  
*identifierhead* = U+10000–U+1FFFD, U+20000–U+2FFFD, U+30000–U+3FFFD, or U+40000–U+4FFFD  
*identifierhead* = U+50000–U+5FFFD, U+60000–U+6FFFD, U+70000–U+7FFFD, or U+80000–U+8FFFD  
*identifierhead* = U+90000–U+9FFFD, U+A0000–U+AFFFD, U+B0000–U+BFFFD, or U+C0000–U+CFFFD  
*identifierhead* = U+D0000–U+DFFFD or U+E0000–U+EFFFD  
  
*identifiercharacter* = Digit 0 through 9  
*identifiercharacter* = U+0300–U+036F, U+1DC0–U+1DFF, U+20D0–U+20FF, or U+FE20–U+FE2F  
*identifiercharacter* = *identifierhead* .  
*identifiercharacters* = *identifiercharacter* *identifiercharacters?* .  
  
*implicitparametername* = **”$“** *decimaldigits* .  
*propertywrapperprojection* = **”$“** *identifiercharacters* .  
*literal* = *numericliteral* | *stringliteral* | *regularexpressionliteral* | *booleanliteral* | *nilliteral* .  
  
*numericliteral* = **”-“***?* *integerliteral* | **”-“***?* *floatingpointliteral* .  
*booleanliteral* = **”true“** | **”false“** .  
*nilliteral* = **”nil“** .  
*integerliteral* = *binaryliteral* .  
*integerliteral* = *octalliteral* .  
*integerliteral* = *decimalliteral* .  
*integerliteral* = *hexadecimalliteral* .  
  
*binaryliteral* = **”0b“** *binarydigit* *binaryliteralcharacters?* .  
*binarydigit* = Digit 0 or 1  
*binaryliteralcharacter* = *binarydigit* | **”\_“** .  
*binaryliteralcharacters* = *binaryliteralcharacter* *binaryliteralcharacters?* .  
  
*octalliteral* = **”0o“** *octaldigit* *octalliteralcharacters?* .  
*octaldigit* = Digit 0 through 7  
*octalliteralcharacter* = *octaldigit* | **”\_“** .  
*octalliteralcharacters* = *octalliteralcharacter* *octalliteralcharacters?* .  
  
*decimalliteral* = *decimaldigit* *decimalliteralcharacters?* .  
*decimaldigit* = Digit 0 through 9  
*decimaldigits* = *decimaldigit* *decimaldigits?* .  
*decimalliteralcharacter* = *decimaldigit* | **”\_“** .  
*decimalliteralcharacters* = *decimalliteralcharacter* *decimalliteralcharacters?* .  
  
*hexadecimalliteral* = **”0x“** *hexadecimaldigit* *hexadecimalliteralcharacters?* .  
*hexadecimaldigit* = Digit 0 through 9, a through f, or A through F  
*hexadecimalliteralcharacter* = *hexadecimaldigit* | **”\_“** .  
*hexadecimalliteralcharacters* = *hexadecimalliteralcharacter* *hexadecimalliteralcharacters?* .  
*floatingpointliteral* = *decimalliteral* *decimalfraction?* *decimalexponent?* .  
*floatingpointliteral* = *hexadecimalliteral* *hexadecimalfraction?* *hexadecimalexponent* .  
  
*decimalfraction* = **”.“** *decimalliteral* .  
*decimalexponent* = *floatingpointe* *sign?* *decimalliteral* .  
  
*hexadecimalfraction* = **”.“** *hexadecimaldigit* *hexadecimalliteralcharacters?* .  
*hexadecimalexponent* = *floatingpointp* *sign?* *decimalliteral* .  
  
*floatingpointe* = **”e“** | **”E“** .  
*floatingpointp* = **”p“** | **”P“** .  
*sign* = **”+“** | **”“** .  
*stringliteral* = *staticstringliteral* | *interpolatedstringliteral* .  
  
*stringliteralopeningdelimiter* = *extendedstringliteraldelimiter?* **”\"“** .  
*stringliteralclosingdelimiter* = **”\"“** *extendedstringliteraldelimiter?* .  
  
*staticstringliteral* = *stringliteralopeningdelimiter* *quotedtext?* *stringliteralclosingdelimiter* .  
*staticstringliteral* = *multilinestringliteralopeningdelimiter* *multilinequotedtext?* *multilinestringliteralclosingdelimiter* .  
  
*multilinestringliteralopeningdelimiter* = *extendedstringliteraldelimiter?* **”\"\"\"“** .  
*multilinestringliteralclosingdelimiter* = **”\"\"\"“** *extendedstringliteraldelimiter?* .  
*extendedstringliteraldelimiter* = **”#“** *extendedstringliteraldelimiter?* .  
  
*quotedtext* = *quotedtextitem* *quotedtext?* .  
*quotedtextitem* = *escapedcharacter* .  
*quotedtextitem* = Any Unicode scalar value except **"**, **\**, U+000A, or U+000D  
  
*multilinequotedtext* = *multilinequotedtextitem* *multilinequotedtext?* .  
*multilinequotedtextitem* = *escapedcharacter* .  
*multilinequotedtextitem* = Any Unicode scalar value except **\**  
*multilinequotedtextitem* = *escapednewline* .  
  
*interpolatedstringliteral* = *stringliteralopeningdelimiter* *interpolatedtext?* *stringliteralclosingdelimiter* .  
*interpolatedstringliteral* = *multilinestringliteralopeningdelimiter* *multilineinterpolatedtext?* *multilinestringliteralclosingdelimiter* .  
  
*interpolatedtext* = *interpolatedtextitem* *interpolatedtext?* .  
*interpolatedtextitem* = **”\(“** *expression* **”)“** | *quotedtextitem* .  
  
*multilineinterpolatedtext* = *multilineinterpolatedtextitem* *multilineinterpolatedtext?* .  
*multilineinterpolatedtextitem* = **”\(“** *expression* **”)“** | *multilinequotedtextitem* .  
  
*escapesequence* = **”\“** *extendedstringliteraldelimiter* .  
*escapedcharacter* = *escapesequence* **”0“** | *escapesequence* **”\“** | *escapesequence* **”t“** | *escapesequence* **”n“** | *escapesequence* **”r“** | *escapesequence* **”"“** | *escapesequence* **”'“** .  
*escapedcharacter* = *escapesequence* **”u“** **”{“** *unicodescalardigits* **”}“** .  
*unicodescalardigits* = Between one and eight hexadecimal digits  
  
*escapednewline* = *escapesequence* *inlinespaces?* *linebreak* .  
*regularexpressionliteral* = *regularexpressionliteralopeningdelimiter* *regularexpression* *regularexpressionliteralclosingdelimiter* .  
*regularexpression* = Any regular expression  
  
*regularexpressionliteralopeningdelimiter* = *extendedregularexpressionliteraldelimiter?* **”/“** .  
*regularexpressionliteralclosingdelimiter* = **”/“** *extendedregularexpressionliteraldelimiter?* .  
  
*extendedregularexpressionliteraldelimiter* = **”#“** *extendedregularexpressionliteraldelimiter?* .  
*operator* = *operatorhead* *operatorcharacters?* .  
*operator* = *dotoperatorhead* *dotoperatorcharacters* .  
  
*operatorhead* = **”/“** | **”=“** | **”“** | **”+“** | **”!“** | **”\*“** | **”%“** | **”<“** | **”>“** | **”&“** | **”|“** | **”^“** | **”~“** | **”?“** .  
*operatorhead* = U+00A1–U+00A7  
*operatorhead* = U+00A9 or U+00AB  
*operatorhead* = U+00AC or U+00AE  
*operatorhead* = U+00B0–U+00B1  
*operatorhead* = U+00B6, U+00BB, U+00BF, U+00D7, or U+00F7  
*operatorhead* = U+2016–U+2017  
*operatorhead* = U+2020–U+2027  
*operatorhead* = U+2030–U+203E  
*operatorhead* = U+2041–U+2053  
*operatorhead* = U+2055–U+205E  
*operatorhead* = U+2190–U+23FF  
*operatorhead* = U+2500–U+2775  
*operatorhead* = U+2794–U+2BFF  
*operatorhead* = U+2E00–U+2E7F  
*operatorhead* = U+3001–U+3003  
*operatorhead* = U+3008–U+3020  
*operatorhead* = U+3030  
  
*operatorcharacter* = *operatorhead* .  
*operatorcharacter* = U+0300–U+036F  
*operatorcharacter* = U+1DC0–U+1DFF  
*operatorcharacter* = U+20D0–U+20FF  
*operatorcharacter* = U+FE00–U+FE0F  
*operatorcharacter* = U+FE20–U+FE2F  
*operatorcharacter* = U+E0100–U+E01EF  
*operatorcharacters* = *operatorcharacter* *operatorcharacters?* .  
  
*dotoperatorhead* = **”.“** .  
*dotoperatorcharacter* = **”.“** | *operatorcharacter* .  
*dotoperatorcharacters* = *dotoperatorcharacter* *dotoperatorcharacters?* .  
  
*infixoperator* = *operator* .  
*prefixoperator* = *operator* .  
*postfixoperator* = *operator* .  
  
*type* = *functiontype* .  
*type* = *arraytype* .  
*type* = *dictionarytype* .  
*type* = *typeidentifier* .  
*type* = *tupletype* .  
*type* = *optionaltype* .  
*type* = *implicitlyunwrappedoptionaltype* .  
*type* = *protocolcompositiontype* .  
*type* = *opaquetype* .  
*type* = *metatypetype* .  
*type* = *anytype* .  
*type* = *selftype* .  
*type* = **”(“** *type* **”)“** .  
*typeannotation* = **”:“** *attributes?* **”inout“***?* *type* .  
*typeidentifier* = *typename* *genericargumentclause?* | *typename* *genericargumentclause?* **”.“** *typeidentifier* .  
*typename* = *identifier* .  
*tupletype* = **”(“** **”)“** | **”(“** *tupletypeelement* **”,“** *tupletypeelementlist* **”)“** .  
*tupletypeelementlist* = *tupletypeelement* | *tupletypeelement* **”,“** *tupletypeelementlist* .  
*tupletypeelement* = *elementname* *typeannotation* | *type* .  
*elementname* = *identifier* .  
*functiontype* = *attributes?* *functiontypeargumentclause* **”async“***?* *throwsclause?* **”>“** *type* .  
  
*functiontypeargumentclause* = **”(“** **”)“** .  
*functiontypeargumentclause* = **”(“** *functiontypeargumentlist* **”...“***?* **”)“** .  
  
*functiontypeargumentlist* = *functiontypeargument* | *functiontypeargument* **”,“** *functiontypeargumentlist* .  
*functiontypeargument* = *attributes?* **”inout“***?* *type* | *argumentlabel* *typeannotation* .  
*argumentlabel* = *identifier* .  
  
*throwsclause* = **”throws“** | **”throws“** **”(“** *type* **”)“** .  
*arraytype* = **”[“** *type* **”]“** .  
*dictionarytype* = **”[“** *type* **”:“** *type* **”]“** .  
*optionaltype* = *type* **”?“** .  
*implicitlyunwrappedoptionaltype* = *type* **”!“** .  
*protocolcompositiontype* = *typeidentifier* **”&“** *protocolcompositioncontinuation* .  
*protocolcompositioncontinuation* = *typeidentifier* | *protocolcompositiontype* .  
*opaquetype* = **”some“** *type* .  
*boxedprotocoltype* = **”any“** *type* .  
*metatypetype* = *type* **”.“** **”Type“** | *type* **”.“** **”Protocol“** .  
*anytype* = **”Any“** .  
*selftype* = **”Self“** .  
*typeinheritanceclause* = **”:“** *typeinheritancelist* .  
*typeinheritancelist* = *attributes?* *typeidentifier* | *attributes?* *typeidentifier* **”,“** *typeinheritancelist* .  
  
*expression* = *tryoperator?* *awaitoperator?* *prefixexpression* *infixexpressions?* \ .  
*prefixexpression* = *prefixoperator?* *postfixexpression* .  
*prefixexpression* = *inoutexpression* .  
*inoutexpression* = **”&“** *primaryexpression* .  
*tryoperator* = **”try“** | **”try“** **”?“** | **”try“** **”!“** .  
*awaitoperator* = **”await“** .  
*infixexpression* = *infixoperator* *prefixexpression* .  
*infixexpression* = *assignmentoperator* *tryoperator?* *awaitoperator?* *prefixexpression* .  
*infixexpression* = *conditionaloperator* *tryoperator?* *awaitoperator?* *prefixexpression* .  
*infixexpression* = *typecastingoperator* .  
*infixexpressions* = *infixexpression* *infixexpressions?* .  
*assignmentoperator* = **”=“** .  
*conditionaloperator* = **”?“** *expression* **”:“** .  
*typecastingoperator* = **”is“** *type* .  
*typecastingoperator* = **”as“** *type* .  
*typecastingoperator* = **”as“** **”?“** *type* .  
*typecastingoperator* = **”as“** **”!“** *type* .  
*primaryexpression* = *identifier* *genericargumentclause?* .  
*primaryexpression* = *literalexpression* .  
*primaryexpression* = *selfexpression* .  
*primaryexpression* = *superclassexpression* .  
*primaryexpression* = *conditionalexpression* .  
*primaryexpression* = *closureexpression* .  
*primaryexpression* = *parenthesizedexpression* .  
*primaryexpression* = *tupleexpression* .  
*primaryexpression* = *implicitmemberexpression* .  
*primaryexpression* = *wildcardexpression* .  
*primaryexpression* = *macroexpansionexpression* .  
*primaryexpression* = *keypathexpression* .  
*primaryexpression* = *selectorexpression* .  
*primaryexpression* = *keypathstringexpression* .  
*literalexpression* = *literal* .  
*literalexpression* = *arrayliteral* | *dictionaryliteral* | *playgroundliteral* .  
  
*arrayliteral* = **”[“** *arrayliteralitems?* **”]“** .  
*arrayliteralitems* = *arrayliteralitem* **”,“***?* | *arrayliteralitem* **”,“** *arrayliteralitems* .  
*arrayliteralitem* = *expression* .  
  
*dictionaryliteral* = **”[“** *dictionaryliteralitems* **”]“** | **”[“** **”:“** **”]“** .  
*dictionaryliteralitems* = *dictionaryliteralitem* **”,“***?* | *dictionaryliteralitem* **”,“** *dictionaryliteralitems* .  
*dictionaryliteralitem* = *expression* **”:“** *expression* .  
  
*playgroundliteral* = **”#colorLiteral“** **”(“** **”red“** **”:“** *expression* **”,“** **”green“** **”:“** *expression* **”,“** **”blue“** **”:“** *expression* **”,“** **”alpha“** **”:“** *expression* **”)“** .  
*playgroundliteral* = **”#fileLiteral“** **”(“** **”resourceName“** **”:“** *expression* **”)“** .  
*playgroundliteral* = **”#imageLiteral“** **”(“** **”resourceName“** **”:“** *expression* **”)“** .  
*selfexpression* = **”self“** | *selfmethodexpression* | *selfsubscriptexpression* | *selfinitializerexpression* .  
  
*selfmethodexpression* = **”self“** **”.“** *identifier* .  
*selfsubscriptexpression* = **”self“** **”[“** *functioncallargumentlist* **”]“** .  
*selfinitializerexpression* = **”self“** **”.“** **”init“** .  
*superclassexpression* = *superclassmethodexpression* | *superclasssubscriptexpression* | *superclassinitializerexpression* .  
  
*superclassmethodexpression* = **”super“** **”.“** *identifier* .  
*superclasssubscriptexpression* = **”super“** **”[“** *functioncallargumentlist* **”]“** .  
*superclassinitializerexpression* = **”super“** **”.“** **”init“** .  
*conditionalexpression* = *ifexpression* | *switchexpression* .  
  
*ifexpression* = **”if“** *conditionlist* **”{“** *statement* **”}“** *ifexpressiontail* .  
*ifexpressiontail* = **”else“** *ifexpression* .  
*ifexpressiontail* = **”else“** **”{“** *statement* **”}“** .  
  
*switchexpression* = **”switch“** *expression* **”{“** *switchexpressioncases* **”}“** .  
*switchexpressioncases* = *switchexpressioncase* *switchexpressioncases?* .  
*switchexpressioncase* = *caselabel* *statement* .  
*switchexpressioncase* = *defaultlabel* *statement* .  
*closureexpression* = **”{“** *attributes?* *closuresignature?* *statements?* **”}“** .  
  
*closuresignature* = *capturelist?* *closureparameterclause* **”async“***?* *throwsclause?* *functionresult?* **”in“** .  
*closuresignature* = *capturelist* **”in“** .  
  
*closureparameterclause* = **”(“** **”)“** | **”(“** *closureparameterlist* **”)“** | *identifierlist* .  
*closureparameterlist* = *closureparameter* | *closureparameter* **”,“** *closureparameterlist* .  
*closureparameter* = *closureparametername* *typeannotation?* .  
*closureparameter* = *closureparametername* *typeannotation* **”...“** .  
*closureparametername* = *identifier* .  
  
*capturelist* = **”[“** *capturelistitems* **”]“** .  
*capturelistitems* = *capturelistitem* | *capturelistitem* **”,“** *capturelistitems* .  
*capturelistitem* = *capturespecifier?* *identifier* .  
*capturelistitem* = *capturespecifier?* *identifier* **”=“** *expression* .  
*capturelistitem* = *capturespecifier?* *selfexpression* .  
*capturespecifier* = **”weak“** | **”unowned“** | **”unowned(safe)“** | **”unowned(unsafe)“** .  
*implicitmemberexpression* = **”.“** *identifier* .  
*implicitmemberexpression* = **”.“** *identifier* **”.“** *postfixexpression* .  
*parenthesizedexpression* = **”(“** *expression* **”)“** .  
*tupleexpression* = **”(“** **”)“** | **”(“** *tupleelement* **”,“** *tupleelementlist* **”)“** .  
*tupleelementlist* = *tupleelement* | *tupleelement* **”,“** *tupleelementlist* .  
*tupleelement* = *expression* | *identifier* **”:“** *expression* .  
*wildcardexpression* = **”\_“** .  
*macroexpansionexpression* = **”#“** *identifier* *genericargumentclause?* *functioncallargumentclause?* *trailingclosures?* .  
*keypathexpression* = **”\“** *type?* **”.“** *keypathcomponents* .  
*keypathcomponents* = *keypathcomponent* | *keypathcomponent* **”.“** *keypathcomponents* .  
*keypathcomponent* = *identifier* *keypathpostfixes?* | *keypathpostfixes* .  
  
*keypathpostfixes* = *keypathpostfix* *keypathpostfixes?* .  
*keypathpostfix* = **”?“** | **”!“** | **”self“** | **”[“** *functioncallargumentlist* **”]“** .  
*selectorexpression* = **”#selector“** **”(“** *expression* **”)“** .  
*selectorexpression* = **”#selector“** **”(“** **”getter:“** *expression* **”)“** .  
*selectorexpression* = **”#selector“** **”(“** **”setter:“** *expression* **”)“** .  
*keypathstringexpression* = **”#keyPath“** **”(“** *expression* **”)“** .  
*postfixexpression* = *primaryexpression* .  
*postfixexpression* = *postfixexpression* *postfixoperator* .  
*postfixexpression* = *functioncallexpression* .  
*postfixexpression* = *initializerexpression* .  
*postfixexpression* = *explicitmemberexpression* .  
*postfixexpression* = *postfixselfexpression* .  
*postfixexpression* = *subscriptexpression* .  
*postfixexpression* = *forcedvalueexpression* .  
*postfixexpression* = *optionalchainingexpression* .  
*functioncallexpression* = *postfixexpression* *functioncallargumentclause* .  
*functioncallexpression* = *postfixexpression* *functioncallargumentclause?* *trailingclosures* .  
  
*functioncallargumentclause* = **”(“** **”)“** | **”(“** *functioncallargumentlist* **”)“** .  
*functioncallargumentlist* = *functioncallargument* | *functioncallargument* **”,“** *functioncallargumentlist* .  
*functioncallargument* = *expression* | *identifier* **”:“** *expression* .  
*functioncallargument* = *operator* | *identifier* **”:“** *operator* .  
  
*trailingclosures* = *closureexpression* *labeledtrailingclosures?* .  
*labeledtrailingclosures* = *labeledtrailingclosure* *labeledtrailingclosures?* .  
*labeledtrailingclosure* = *identifier* **”:“** *closureexpression* .  
*initializerexpression* = *postfixexpression* **”.“** **”init“** .  
*initializerexpression* = *postfixexpression* **”.“** **”init“** **”(“** *argumentnames* **”)“** .  
*explicitmemberexpression* = *postfixexpression* **”.“** *decimaldigits* .  
*explicitmemberexpression* = *postfixexpression* **”.“** *identifier* *genericargumentclause?* .  
*explicitmemberexpression* = *postfixexpression* **”.“** *identifier* **”(“** *argumentnames* **”)“** .  
*explicitmemberexpression* = *postfixexpression* *conditionalcompilationblock* .  
  
*argumentnames* = *argumentname* *argumentnames?* .  
*argumentname* = *identifier* **”:“** .  
*postfixselfexpression* = *postfixexpression* **”.“** **”self“** .  
*subscriptexpression* = *postfixexpression* **”[“** *functioncallargumentlist* **”]“** .  
*forcedvalueexpression* = *postfixexpression* **”!“** .  
*optionalchainingexpression* = *postfixexpression* **”?“** .  
  
*statement* = *expression* **”;“***?* .  
*statement* = *declaration* **”;“***?* .  
*statement* = *loopstatement* **”;“***?* .  
*statement* = *branchstatement* **”;“***?* .  
*statement* = *labeledstatement* **”;“***?* .  
*statement* = *controltransferstatement* **”;“***?* .  
*statement* = *deferstatement* **”;“***?* .  
*statement* = *dostatement* **”;“***?* .  
*statement* = *compilercontrolstatement* .  
*statements* = *statement* *statements?* .  
*loopstatement* = *forinstatement* .  
*loopstatement* = *whilestatement* .  
*loopstatement* = *repeatwhilestatement* .  
*forinstatement* = **”for“** **”case“***?* *pattern* **”in“** *expression* *whereclause?* *codeblock* .  
*whilestatement* = **”while“** *conditionlist* *codeblock* .  
  
*conditionlist* = *condition* | *condition* **”,“** *conditionlist* .  
*condition* = *expression* | *availabilitycondition* | *casecondition* | *optionalbindingcondition* .  
  
*casecondition* = **”case“** *pattern* *initializer* .  
*optionalbindingcondition* = **”let“** *pattern* *initializer?* | **”var“** *pattern* *initializer?* .  
*repeatwhilestatement* = **”repeat“** *codeblock* **”while“** *expression* .  
*branchstatement* = *ifstatement* .  
*branchstatement* = *guardstatement* .  
*branchstatement* = *switchstatement* .  
*ifstatement* = **”if“** *conditionlist* *codeblock* *elseclause?* .  
*elseclause* = **”else“** *codeblock* | **”else“** *ifstatement* .  
*guardstatement* = **”guard“** *conditionlist* **”else“** *codeblock* .  
*switchstatement* = **”switch“** *expression* **”{“** *switchcases?* **”}“** .  
*switchcases* = *switchcase* *switchcases?* .  
*switchcase* = *caselabel* *statements* .  
*switchcase* = *defaultlabel* *statements* .  
*switchcase* = *conditionalswitchcase* .  
  
*caselabel* = *attributes?* **”case“** *caseitemlist* **”:“** .  
*caseitemlist* = *pattern* *whereclause?* | *pattern* *whereclause?* **”,“** *caseitemlist* .  
*defaultlabel* = *attributes?* **”default“** **”:“** .  
  
*whereclause* = **”where“** *whereexpression* .  
*whereexpression* = *expression* .  
  
*conditionalswitchcase* = *switchifdirectiveclause* *switchelseifdirectiveclauses?* *switchelsedirectiveclause?* *endifdirective* .  
*switchifdirectiveclause* = *ifdirective* *compilationcondition* *switchcases?* .  
*switchelseifdirectiveclauses* = *elseifdirectiveclause* *switchelseifdirectiveclauses?* .  
*switchelseifdirectiveclause* = *elseifdirective* *compilationcondition* *switchcases?* .  
*switchelsedirectiveclause* = *elsedirective* *switchcases?* .  
*labeledstatement* = *statementlabel* *loopstatement* .  
*labeledstatement* = *statementlabel* *ifstatement* .  
*labeledstatement* = *statementlabel* *switchstatement* .  
*labeledstatement* = *statementlabel* *dostatement* .  
  
*statementlabel* = *labelname* **”:“** .  
*labelname* = *identifier* .  
*controltransferstatement* = *breakstatement* .  
*controltransferstatement* = *continuestatement* .  
*controltransferstatement* = *fallthroughstatement* .  
*controltransferstatement* = *returnstatement* .  
*controltransferstatement* = *throwstatement* .  
*breakstatement* = **”break“** *labelname?* .  
*continuestatement* = **”continue“** *labelname?* .  
*fallthroughstatement* = **”fallthrough“** .  
*returnstatement* = **”return“** *expression?* .  
*throwstatement* = **”throw“** *expression* .  
*deferstatement* = **”defer“** *codeblock* .  
*dostatement* = **”do“** *throwsclause?* *codeblock* *catchclauses?* .  
*catchclauses* = *catchclause* *catchclauses?* .  
*catchclause* = **”catch“** *catchpatternlist?* *codeblock* .  
*catchpatternlist* = *catchpattern* | *catchpattern* **”,“** *catchpatternlist* .  
*catchpattern* = *pattern* *whereclause?* .  
*compilercontrolstatement* = *conditionalcompilationblock* .  
*compilercontrolstatement* = *linecontrolstatement* .  
*compilercontrolstatement* = *diagnosticstatement* .  
*conditionalcompilationblock* = *ifdirectiveclause* *elseifdirectiveclauses?* *elsedirectiveclause?* *endifdirective* .  
  
*ifdirectiveclause* = *ifdirective* *compilationcondition* *statements?* .  
*elseifdirectiveclauses* = *elseifdirectiveclause* *elseifdirectiveclauses?* .  
*elseifdirectiveclause* = *elseifdirective* *compilationcondition* *statements?* .  
*elsedirectiveclause* = *elsedirective* *statements?* .  
*ifdirective* = **”#if“** .  
*elseifdirective* = **”#elseif“** .  
*elsedirective* = **”#else“** .  
*endifdirective* = **”#endif“** .  
  
*compilationcondition* = *platformcondition* .  
*compilationcondition* = *identifier* .  
*compilationcondition* = *booleanliteral* .  
*compilationcondition* = **”(“** *compilationcondition* **”)“** .  
*compilationcondition* = **”!“** *compilationcondition* .  
*compilationcondition* = *compilationcondition* **”&&“** *compilationcondition* .  
*compilationcondition* = *compilationcondition* **”||“** *compilationcondition* .  
  
*platformcondition* = **”os“** **”(“** *operatingsystem* **”)“** .  
*platformcondition* = **”arch“** **”(“** *architecture* **”)“** .  
*platformcondition* = **”swift“** **”(“** **”>=“** *swiftversion* **”)“** | **”swift“** **”(“** **”<“** *swiftversion* **”)“** .  
*platformcondition* = **”compiler“** **”(“** **”>=“** *swiftversion* **”)“** | **”compiler“** **”(“** **”<“** *swiftversion* **”)“** .  
*platformcondition* = **”canImport“** **”(“** *importpath* **”)“** .  
*platformcondition* = **”targetEnvironment“** **”(“** *environment* **”)“** .  
  
*operatingsystem* = **”macOS“** | **”iOS“** | **”watchOS“** | **”tvOS“** | **”visionOS“** | **”Linux“** | **”Windows“** .  
*architecture* = **”i386“** | **”x86\_64“** | **”arm“** | **”arm64“** .  
*swiftversion* = *decimaldigits* *swiftversioncontinuation?* .  
*swiftversioncontinuation* = **”.“** *decimaldigits* *swiftversioncontinuation?* .  
*environment* = **”simulator“** | **”macCatalyst“** .  
*linecontrolstatement* = **”#sourceLocation“** **”(“** **”file:“** *filepath* **”,“** **”line:“** *linenumber* **”)“** .  
*linecontrolstatement* = **”#sourceLocation“** **”(“** **”)“** .  
*linenumber* = A decimal integer greater than zero .  
*filepath* = *staticstringliteral* .  
*availabilitycondition* = **”#available“** **”(“** *availabilityarguments* **”)“** .  
*availabilitycondition* = **”#unavailable“** **”(“** *availabilityarguments* **”)“** .  
*availabilityarguments* = *availabilityargument* | *availabilityargument* **”,“** *availabilityarguments* .  
*availabilityargument* = *platformname* *platformversion* .  
*availabilityargument* = **”\*“** .  
  
*platformname* = **”iOS“** | **”iOSApplicationExtension“** .  
*platformname* = **”macOS“** | **”macOSApplicationExtension“** .  
*platformname* = **”macCatalyst“** | **”macCatalystApplicationExtension“** .  
*platformname* = **”watchOS“** | **”watchOSApplicationExtension“** .  
*platformname* = **”tvOS“** | **”tvOSApplicationExtension“** .  
*platformname* = **”visionOS“** | **”visionOSApplicationExtension“** .  
*platformversion* = *decimaldigits* .  
*platformversion* = *decimaldigits* **”.“** *decimaldigits* .  
*platformversion* = *decimaldigits* **”.“** *decimaldigits* **”.“** *decimaldigits* .  
  
*declaration* = *importdeclaration* .  
*declaration* = *constantdeclaration* .  
*declaration* = *variabledeclaration* .  
*declaration* = *typealiasdeclaration* .  
*declaration* = *functiondeclaration* .  
*declaration* = *enumdeclaration* .  
*declaration* = *structdeclaration* .  
*declaration* = *classdeclaration* .  
*declaration* = *actordeclaration* .  
*declaration* = *protocoldeclaration* .  
*declaration* = *initializerdeclaration* .  
*declaration* = *deinitializerdeclaration* .  
*declaration* = *extensiondeclaration* .  
*declaration* = *subscriptdeclaration* .  
*declaration* = *operatordeclaration* .  
*declaration* = *precedencegroupdeclaration .* \// WTF is \ ???  
*topleveldeclaration* = *statements?* .  
*codeblock* = **”{“** *statements?* **”}“** .  
*importdeclaration* = *attributes?* **”import“** *importkind?* *importpath* .  
  
*importkind* = **”typealias“** | **”struct“** | **”class“** | **”enum“** | **”protocol“** | **”let“** | **”var“** | **”func“** .  
*importpath* = *identifier* | *identifier* **”.“** *importpath* .  
*constantdeclaration* = *attributes?* *declarationmodifiers?* **”let“** *patterninitializerlist* .  
  
*patterninitializerlist* = *patterninitializer* | *patterninitializer* **”,“** *patterninitializerlist* .  
*patterninitializer* = *pattern* *initializer?* .  
*initializer* = **”=“** *expression* .  
*variabledeclaration* = *variabledeclarationhead* *patterninitializerlist* .  
*variabledeclaration* = *variabledeclarationhead* *variablename* *typeannotation* *codeblock* .  
*variabledeclaration* = *variabledeclarationhead* *variablename* *typeannotation* *gettersetterblock* .  
*variabledeclaration* = *variabledeclarationhead* *variablename* *typeannotation* *gettersetterkeywordblock* .  
*variabledeclaration* = *variabledeclarationhead* *variablename* *initializer* *willSetdidSetblock* .  
*variabledeclaration* = *variabledeclarationhead* *variablename* *typeannotation* *initializer?* *willSetdidSetblock* .  
  
*variabledeclarationhead* = *attributes?* *declarationmodifiers?* **”var“** .  
*variablename* = *identifier* .  
  
*gettersetterblock* = *codeblock* .  
*gettersetterblock* = **”{“** *getterclause* *setterclause?* **”}“** .  
*gettersetterblock* = **”{“** *setterclause* *getterclause* **”}“** .  
*getterclause* = *attributes?* *mutationmodifier?* **”get“** *codeblock* .  
*setterclause* = *attributes?* *mutationmodifier?* **”set“** *settername?* *codeblock* .  
*settername* = **”(“** *identifier* **”)“** .  
  
*gettersetterkeywordblock* = **”{“** *getterkeywordclause* *setterkeywordclause?* **”}“** .  
*gettersetterkeywordblock* = **”{“** *setterkeywordclause* *getterkeywordclause* **”}“** .  
*getterkeywordclause* = *attributes?* *mutationmodifier?* **”get“** .  
*setterkeywordclause* = *attributes?* *mutationmodifier?* **”set“** .  
  
*willSetdidSetblock* = **”{“** *willSetclause* *didSetclause?* **”}“** .  
*willSetdidSetblock* = **”{“** *didSetclause* *willSetclause?* **”}“** .  
*willSetclause* = *attributes?* **”willSet“** *settername?* *codeblock* .  
*didSetclause* = *attributes?* **”didSet“** *settername?* *codeblock* .  
*typealiasdeclaration* = *attributes?* *accesslevelmodifier?* **”typealias“** *typealiasname* *genericparameterclause?* *typealiasassignment* .  
*typealiasname* = *identifier* .  
*typealiasassignment* = **”=“** *type* .  
*functiondeclaration* = *functionhead* *functionname* *genericparameterclause?* *functionsignature* *genericwhereclause?* *functionbody?* .  
  
*functionhead* = *attributes?* *declarationmodifiers?* **”func“** .  
*functionname* = *identifier* | *operator* .  
  
*functionsignature* = *parameterclause* **”async“***?* *throwsclause?* *functionresult?* .  
*functionsignature* = *parameterclause* **”async“***?* **”rethrows“** *functionresult?* .  
*functionresult* = **”>“** *attributes?* *type* .  
*functionbody* = *codeblock* .  
  
*parameterclause* = **”(“** **”)“** | **”(“** *parameterlist* **”)“** .  
*parameterlist* = *parameter* | *parameter* **”,“** *parameterlist* .  
*parameter* = *externalparametername?* *localparametername* *parametertypeannotation* *defaultargumentclause?* .  
*parameter* = *externalparametername?* *localparametername* *parametertypeannotation* .  
*parameter* = *externalparametername?* *localparametername* *parametertypeannotation* **”...“** .  
  
*externalparametername* = *identifier* .  
*localparametername* = *identifier* .  
*parametertypeannotation* = **”:“** *attributes?* *parametermodifier?* *type* .  
*parametermodifier* = **”inout“** | **”borrowing“** | **”consuming“** *defaultargumentclause* = **”=“** *expression* .  
*enumdeclaration* = *attributes?* *accesslevelmodifier?* *unionstyleenum* .  
*enumdeclaration* = *attributes?* *accesslevelmodifier?* *rawvaluestyleenum* .  
  
*unionstyleenum* = **”indirect“***?* **”enum“** *enumname* *genericparameterclause?* *typeinheritanceclause?* *genericwhereclause?* **”{“** *unionstyleenummembers?* **”}“** .  
*unionstyleenummembers* = *unionstyleenummember* *unionstyleenummembers?* .  
*unionstyleenummember* = *declaration* | *unionstyleenumcaseclause* | *compilercontrolstatement* .  
*unionstyleenumcaseclause* = *attributes?* **”indirect“***?* **”case“** *unionstyleenumcaselist* .  
*unionstyleenumcaselist* = *unionstyleenumcase* | *unionstyleenumcase* **”,“** *unionstyleenumcaselist* .  
*unionstyleenumcase* = *enumcasename* *tupletype?* .  
*enumname* = *identifier* .  
*enumcasename* = *identifier* .  
  
*rawvaluestyleenum* = **”enum“** *enumname* *genericparameterclause?* *typeinheritanceclause* *genericwhereclause?* **”{“** *rawvaluestyleenummembers* **”}“** .  
*rawvaluestyleenummembers* = *rawvaluestyleenummember* *rawvaluestyleenummembers?* .  
*rawvaluestyleenummember* = *declaration* | *rawvaluestyleenumcaseclause* | *compilercontrolstatement* .  
*rawvaluestyleenumcaseclause* = *attributes?* **”case“** *rawvaluestyleenumcaselist* .  
*rawvaluestyleenumcaselist* = *rawvaluestyleenumcase* | *rawvaluestyleenumcase* **”,“** *rawvaluestyleenumcaselist* .  
*rawvaluestyleenumcase* = *enumcasename* *rawvalueassignment?* .  
*rawvalueassignment* = **”=“** *rawvalueliteral* .  
*rawvalueliteral* = *numericliteral* | *staticstringliteral* | *booleanliteral* .  
*structdeclaration* = *attributes?* *accesslevelmodifier?* **”struct“** *structname* *genericparameterclause?* *typeinheritanceclause?* *genericwhereclause?* *structbody* .  
*structname* = *identifier* .  
*structbody* = **”{“** *structmembers?* **”}“** .  
  
*structmembers* = *structmember* *structmembers?* .  
*structmember* = *declaration* | *compilercontrolstatement* .  
*classdeclaration* = *attributes?* *accesslevelmodifier?* **”final“***?* **”class“** *classname* *genericparameterclause?* *typeinheritanceclause?* *genericwhereclause?* *classbody* .  
*classdeclaration* = *attributes?* **”final“** *accesslevelmodifier?* **”class“** *classname* *genericparameterclause?* *typeinheritanceclause?* *genericwhereclause?* *classbody* .  
*classname* = *identifier* .  
*classbody* = **”{“** *classmembers?* **”}“** .  
  
*classmembers* = *classmember* *classmembers?* .  
*classmember* = *declaration* | *compilercontrolstatement* .  
*actordeclaration* = *attributes?* *accesslevelmodifier?* **”actor“** *actorname* *genericparameterclause?* *typeinheritanceclause?* *genericwhereclause?* *actorbody* .  
*actorname* = *identifier* .  
*actorbody* = **”{“** *actormembers?* **”}“** .  
  
*actormembers* = *actormember* *actormembers?* .  
*actormember* = *declaration* | *compilercontrolstatement* .  
*protocoldeclaration* = *attributes?* *accesslevelmodifier?* **”protocol“** *protocolname* *typeinheritanceclause?* *genericwhereclause?* *protocolbody* .  
*protocolname* = *identifier* .  
*protocolbody* = **”{“** *protocolmembers?* **”}“** .  
  
*protocolmembers* = *protocolmember* *protocolmembers?* .  
*protocolmember* = *protocolmemberdeclaration* | *compilercontrolstatement* .  
  
*protocolmemberdeclaration* = *protocolpropertydeclaration* .  
*protocolmemberdeclaration* = *protocolmethoddeclaration* .  
*protocolmemberdeclaration* = *protocolinitializerdeclaration* .  
*protocolmemberdeclaration* = *protocolsubscriptdeclaration* .  
*protocolmemberdeclaration* = *protocolassociatedtypedeclaration* .  
*protocolmemberdeclaration* = *typealiasdeclaration* .  
*protocolpropertydeclaration* = *variabledeclarationhead* *variablename* *typeannotation* *gettersetterkeywordblock* .  
*protocolmethoddeclaration* = *functionhead* *functionname* *genericparameterclause?* *functionsignature* *genericwhereclause?* .  
*protocolinitializerdeclaration* = *initializerhead* *genericparameterclause?* *parameterclause* *throwsclause?* *genericwhereclause?* .  
*protocolinitializerdeclaration* = *initializerhead* *genericparameterclause?* *parameterclause* **”rethrows“** *genericwhereclause?* .  
*protocolsubscriptdeclaration* = *subscripthead* *subscriptresult* *genericwhereclause?* *gettersetterkeywordblock* .  
*protocolassociatedtypedeclaration* = *attributes?* *accesslevelmodifier?* **”associatedtype“** *typealiasname* *typeinheritanceclause?* *typealiasassignment?* *genericwhereclause?* .  
*initializerdeclaration* = *initializerhead* *genericparameterclause?* *parameterclause* **”async“***?* *throwsclause?* *genericwhereclause?* *initializerbody* .  
*initializerdeclaration* = *initializerhead* *genericparameterclause?* *parameterclause* **”async“***?* **”rethrows“** *genericwhereclause?* *initializerbody* .  
*initializerhead* = *attributes?* *declarationmodifiers?* **”init“** .  
*initializerhead* = *attributes?* *declarationmodifiers?* **”init“** **”?“** .  
*initializerhead* = *attributes?* *declarationmodifiers?* **”init“** **”!“** .  
*initializerbody* = *codeblock* .  
*deinitializerdeclaration* = *attributes?* **”deinit“** *codeblock* .  
*extensiondeclaration* = *attributes?* *accesslevelmodifier?* **”extension“** *typeidentifier* *typeinheritanceclause?* *genericwhereclause?* *extensionbody* .  
*extensionbody* = **”{“** *extensionmembers?* **”}“** .  
  
*extensionmembers* = *extensionmember* *extensionmembers?* .  
*extensionmember* = *declaration* | *compilercontrolstatement* .  
*subscriptdeclaration* = *subscripthead* *subscriptresult* *genericwhereclause?* *codeblock* .  
*subscriptdeclaration* = *subscripthead* *subscriptresult* *genericwhereclause?* *gettersetterblock* .  
*subscriptdeclaration* = *subscripthead* *subscriptresult* *genericwhereclause?* *gettersetterkeywordblock* .  
*subscripthead* = *attributes?* *declarationmodifiers?* **”subscript“** *genericparameterclause?* *parameterclause* .  
*subscriptresult* = **”>“** *attributes?* *type* .  
*macrodeclaration* = *macrohead* *identifier* *genericparameterclause?* *macrosignature* *macrodefinition?* *genericwhereclause* .  
*macrohead* = *attributes?* *declarationmodifiers?* **”macro“** .  
*macrosignature* = *parameterclause* *macrofunctionsignatureresult?* .  
*macrofunctionsignatureresult* = **”>“** *type* .  
*macrodefinition* = **”=“** *expression* .  
*operatordeclaration* = *prefixoperatordeclaration* | *postfixoperatordeclaration* | *infixoperatordeclaration* .  
  
*prefixoperatordeclaration* = **”prefix“** **”operator“** *operator* .  
*postfixoperatordeclaration* = **”postfix“** **”operator“** *operator* .  
*infixoperatordeclaration* = **”infix“** **”operator“** *operator* *infixoperatorgroup?* .  
  
*infixoperatorgroup* = **”:“** *precedencegroupname* .  
*precedencegroupdeclaration* = **”precedencegroup“** *precedencegroupname* **”{“** *precedencegroupattributes?* **”}“** .  
  
*precedencegroupattributes* = *precedencegroupattribute* *precedencegroupattributes?* .  
*precedencegroupattribute* = *precedencegrouprelation* .  
*precedencegroupattribute* = *precedencegroupassignment* .  
*precedencegroupattribute* = *precedencegroupassociativity* .  
  
*precedencegrouprelation* = **”higherThan“** **”:“** *precedencegroupnames* .  
*precedencegrouprelation* = **”lowerThan“** **”:“** *precedencegroupnames* .  
  
*precedencegroupassignment* = **”assignment“** **”:“** *booleanliteral* .  
  
*precedencegroupassociativity* = **”associativity“** **”:“** **”left“** .  
*precedencegroupassociativity* = **”associativity“** **”:“** **”right“** .  
*precedencegroupassociativity* = **”associativity“** **”:“** **”none“** .  
  
*precedencegroupnames* = *precedencegroupname* | *precedencegroupname* **”,“** *precedencegroupnames* .  
*precedencegroupname* = *identifier* .  
*declarationmodifier* = **”class“** | **”convenience“** | **”dynamic“** | **”final“** | **”infix“** | **”lazy“** | **”optional“** | **”override“** | **”postfix“** | **”prefix“** | **”required“** | **”static“** | **”unowned“** | **”unowned“** **”(“** **”safe“** **”)“** | **”unowned“** **”(“** **”unsafe“** **”)“** | **”weak“** .  
*declarationmodifier* = *accesslevelmodifier* .  
*declarationmodifier* = *mutationmodifier* .  
*declarationmodifier* = *actorisolationmodifier* .  
*declarationmodifiers* = *declarationmodifier* *declarationmodifiers?* .  
  
*accesslevelmodifier* = **”private“** | **”private“** **”(“** **”set“** **”)“** .  
*accesslevelmodifier* = **”fileprivate“** | **”fileprivate“** **”(“** **”set“** **”)“** .  
*accesslevelmodifier* = **”internal“** | **”internal“** **”(“** **”set“** **”)“** .  
*accesslevelmodifier* = **”package“** | **”package“** **”(“** **”set“** **”)“** .  
*accesslevelmodifier* = **”public“** | **”public“** **”(“** **”set“** **”)“** .  
*accesslevelmodifier* = **”open“** | **”open“** **”(“** **”set“** **”)“** .  
  
*mutationmodifier* = **”mutating“** | **”nonmutating“** .  
  
*actorisolationmodifier* = **”nonisolated“** .  
  
*attribute* = **”@“** *attributename* *attributeargumentclause?* .  
*attributename* = *identifier* .  
*attributeargumentclause* = **”(“** *balancedtokens?* **”)“** .  
*attributes* = *attribute* *attributes?* .  
  
*balancedtokens* = *balancedtoken* *balancedtokens?* .  
*balancedtoken* = **”(“** *balancedtokens?* **”)“** .  
*balancedtoken* = **”[“** *balancedtokens?* **”]“** .  
*balancedtoken* = **”{“** *balancedtokens?* **”}“** .  
*balancedtoken* = Any identifier, keyword, literal, or operator .  
*balancedtoken* = Any punctuation except **(**, **)**, **[**, **]**, **{**, or **}** .  
  
*pattern* = *wildcardpattern* *typeannotation?* .  
*pattern* = *identifierpattern* *typeannotation?* .  
*pattern* = *valuebindingpattern* .  
*pattern* = *tuplepattern* *typeannotation?* .  
*pattern* = *enumcasepattern* .  
*pattern* = *optionalpattern* .  
*pattern* = *typecastingpattern* .  
*pattern* = *expressionpattern* .  
*wildcardpattern* = **”\_“** .  
*identifierpattern* = *identifier* .  
*valuebindingpattern* = **”var“** *pattern* | **”let“** *pattern* .  
*tuplepattern* = **”(“** *tuplepatternelementlist?* **”)“** .  
*tuplepatternelementlist* = *tuplepatternelement* | *tuplepatternelement* **”,“** *tuplepatternelementlist* .  
*tuplepatternelement* = *pattern* | *identifier* **”:“** *pattern* .  
*enumcasepattern* = *typeidentifier?* **”.“** *enumcasename* *tuplepattern?* .  
*optionalpattern* = *identifierpattern* **”?“** .  
*typecastingpattern* = *ispattern* | *aspattern* .  
*ispattern* = **”is“** *type* .  
*aspattern* = *pattern* **”as“** *type* .  
*expressionpattern* = *expression* .  
  
*genericparameterclause* = **”<“** *genericparameterlist* **”>“** .  
*genericparameterlist* = *genericparameter* | *genericparameter* **”,“** *genericparameterlist* .  
*genericparameter* = *typename* .  
*genericparameter* = *typename* **”:“** *typeidentifier* .  
*genericparameter* = *typename* **”:“** *protocolcompositiontype* .  
  
*genericwhereclause* = **”where“** *requirementlist* .  
*requirementlist* = *requirement* | *requirement* **”,“** *requirementlist* .  
*requirement* = *conformancerequirement* | *sametyperequirement* .  
  
*conformancerequirement* = *typeidentifier* **”:“** *typeidentifier* .  
*conformancerequirement* = *typeidentifier* **”:“** *protocolcompositiontype* .  
*sametyperequirement* = *typeidentifier* **”==“** *type* .  
*genericargumentclause* = **”<“** *genericargumentlist* **”>“** .  
*genericargumentlist* = *genericargument* | *genericargument* **”,“** *genericargumentlist* .  
*genericargument* = *type* .