### Java Summary Sheet for Mason High School APCS Classes

#### Control Structures

A simple if..else statement

if ( a > b )  
 System.out.println( a + " is larger than " + b );  
 else  
 System.out.println( b + " is larger than " + a );

Checking a range

// is x in range low .. high?  
 if ( low <= x && x >= high )  
 {  
 System.out.println ( "in range" );  
 }  
 else  
 {  
 System.out.println ( "out of range" );  
 }

An if statement with multiple expressions and an else statement.

if (testscore >= 90) {  
 grade = 'A';  
} else if (testscore >= 80) {  
 grade = 'B';  
} else if (testscore >= 70) {  
 grade = 'C';  
} else if (testscore >= 60) {  
 grade = 'D';  
} else {  
 grade = 'F';  
}

Switch/Case

// n should be char or int  
 switch ( n )  
 {  
 case 1:  
 System.out.println ( "one" );  
 break;  
 case 2:  
 System.out.println ( "two" );  
 break;  
 default:  
 System.out.println ( "something else" );  
 }

#### Loops

For loop

for ( int i = 0; i <= n; i++ )  
 {  
 System.out.println( i );  
 }

Enumeration

for ( Enumeration e = props.propertyNames(); e.hasMoreElements(); )  
 {  
 String key = (String) e.nextElement();  
 System.out.println ( key );  
 }

Iterator

for ( Iterator iter = list.iterator(); iter.hasNext(); )  
 {  
 String key = (String) iter.next();  
 System.out.println( key );  
 }

While

while ( moreData() )  
 {  
 readIt();  
 }

Do/While

do  
 {  
 readIt();  
 if ( done() )   
 break;  
 if ( bypassThisOne() )   
 continue;  
 processIt();  
 }  
 while ( moreData() );

#### Try/Catch/Throw

public class Test  
 {  
 public static void main (String [] args)  
 {  
 try {  
 dangerMethod();  
 }  
 catch ( StrangeException e ) {  
 System.out.println ( "exception" + e.getMessage() );  
 }  
 }  
   
 void dangerMethod() throws StrangeException  
 {  
 if ( unexpected() ) throw new StrangeException ( "from dangerMethod" ) ;  
 }   
 }

#### Primitives (used in our course)

| Type | Signed? | Bits | Bytes | Lowest | Highest |
| --- | --- | --- | --- | --- | --- |
| boolean | n/a | 1 | 1 | false | true |
| char | unsigned Unicode | 16 | 2 | '\u0000' | '\uffff' |
| int | signed | 32 | 4 | -2,147,483,648 | +2,147,483,647 |
| long | signed | 64 | 8 | -9,223,372,036,854,775,808 | +9,223,372,036,854,775,807 |
| double | signed exponent and mantissa | 64 | 8 | ±4.94065645841246544e-324 | ±1.79769313486231570e+308 |

#### Keywords

| Java Keywords | | | | |
| --- | --- | --- | --- | --- |
| abstract | do | import | public | throws |
| boolean | double | instanceof | return | transient |
| break | else | int | short | try |
| byte | extends | interface | static | void |
| case | final | long | strictfp | volatile |
| catch | finally | native | super | while |
| char | float | new | switch |  |
| class | for | package | synchroni**z**ed |  |
| continue | if | private | this |  |
| default | implements | protected | throw |  |
| Reserved keywords (not currently in use) | | | | |
| const | goto |  |  |  |
| Reserved Literals | | | | |
| null | true | false |  |  |

#### Precedence (in descending priority order)

| Operator | Association |
| --- | --- |
| postfix operators | [] . (*params*) *expr*++ *expr*-- |
| unary operators | ++*expr* --*expr* +*expr* -*expr* ~ ! |
| creation or cast | new (*type*)*expr* |
| multiplicative | \* / % |
| additive | + - |
| shift | << >> >>> |
| relational | < > <= >= instanceof |
| equality | == != |
| bitwise AND | & |
| bitwise exclusive OR | ^ |
| bitwise inclusive OR | | |
| logical AND | && |
| logical OR | || |
| conditional | ? : |
| assignment | = += -= \*= /= %= &= ^= |= <<= >>= >>>= |