Invariant: a statement whose truth can be checked at any point in true

Loop Invariant: a statement that is true when entering a loop and every time after that as well

Void insertionsort (int a [], int length) {

Int i;

For (I = 0, I < length, i++) {

//insert a[i] into sorted list

Int j;

Int v = a[i];

For(j = length – 1, j >= 0, j--) {

If (a[j] <= v) break;

A[j+1] = a[i]

}

A[j+1] = v;

}

}

Abstract Data Types

Contracts

Polymorphic Containers

Higher order map functions

Parametric Polymorphism

#define T Table \_T

Typdef struct T \*T

Extern T Table\_new (int hint, int cmp(const void \*x, const void \*y), unsigned hash (const void \*key))

Extern void \*Table\_put(T table, const void \*key, void \*value);

Extern void \*Table\_get(T table, const void \*key);

Void \* is an unknown type or unknown pointer type

Function closure

Hansons Unboxed arrays

Array\_T – unboxed

List\_T, Seq\_T – boxed

Typedef struct T \*T;

T Array\_new(int length, int size);

Void Array\_free(T \*array);

Int Array\_length(T array);

Int Array\_size(T array);

Void \*Array\_get(T array);

DO NOT USE void \*Array\_put(T array, int I, void \*elem);

Void Array\_rezie(T array, int length);

T Array\_copy(T array, int length);

Void \* points to the VALUE owned by the array in unboxed

Void \* is the value in boxed

Struct pixel \*p = Array\_get(a, i);

Assert(sizeof (\*p) == Array\_size(a));

(\*p).red = 0;

P ->red = 0;

\*p = darkened\_pixel(); -> make function darkened\_pixel

Void \* Array\_get(T array, int i);

Struct thing \*p = Array\_get(a,i);

Sudoku

Import

Stdlib.h -> exit

Stdio.h -> fprintf

Assert.h -> assert

Pnmrdr.h -> pnmrdr

Uarray2.h->uarray2\_new and other

Uarray2.c -> implements things declared in uarray2.h (all same functions)

Uarray2.c import -> array.h ->Array\_new()

Uarray2new()

Testing -> sh –x compile

gcc –c uarray2.c –I /usr/local/include –I/usr/local/cii/include

gcc –o Sudoku uarray2.o sudoko.o -;pnmrdr –lc –lcii –L/usr/local/lib –L/usr/local/cii/lib