Christiaan Cronje

COSC3319.01

8 A.M.

April 20, 2017

Lab 2 Grading Option “A”

**Program.cs**

using System;

using System.IO;

namespace MultiStack

{

class Program

{

static void Main(string[] args)

{

// Get first stack size info

int m = 0, n = 0, L0 = 0, LB = 0, UB = 0;

Console.Write("Enter a Lower Bound: ");

LB = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter a Upper Bound: ");

UB = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the stack start index: ");

L0 = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the total amount of space for the stacks: ");

m = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the number of stacks: ");

n = Convert.ToInt32(Console.ReadLine());

// Create string type Super Stack

SuperStack<string> cbDataStack = SuperStack<string>.CreateStack(LB, UB, L0, m, n);

// Get C/B option input file information

string fileName = "";

do

{

Console.Write("Enter name of input file for C/B data: ");

fileName = Console.ReadLine();

if (!File.Exists(fileName))

{

Console.WriteLine("File not found");

}

}

while (!File.Exists(fileName));

string[] input = File.ReadAllLines(fileName);

// Process C/B data and push/pop into stack

int s = 0;

string o = "";

for (int i = 0; i < input.Length; i++)

{

if (input[i].Substring(0, 1).Equals("I"))

{

s = Convert.ToInt32(input[i].Substring(1, 2)) - 1;

o = input[i].Substring(3);

// If push cannot be completed, even after any reallocation

// then stop processing, stack is full

if (!cbDataStack.Push(s, o))

break;

}

else if (input[i].Substring(0, 1).Equals("D"))

{

s = Convert.ToInt32(input[i].Substring(1)) - 1;

cbDataStack.Pop(s);

}

}

Console.Write("Final Stack Result for C/B Data: ");

cbDataStack.PrintStack();

// Get second stack size info

Console.Write("Enter a Lower Bound: ");

LB = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter a Upper Bound: ");

UB = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the stack start index: ");

L0 = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the total amount of space for the stacks: ");

m = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the number of stacks: ");

n = Convert.ToInt32(Console.ReadLine());

// Create custom type Super Stack

SuperStack <Date> aDataStack = SuperStack<Date>.CreateStack(LB, UB, L0, m, n);

// Get A option input file information

fileName = "";

input = new string[0];

do

{

Console.Write("Enter name of input file for A data: ");

fileName = Console.ReadLine();

if (!File.Exists(fileName))

{

Console.WriteLine("File not found");

}

}

while (!File.Exists(fileName));

input = File.ReadAllLines(fileName);

// Parse input to custom class/struct

// Push/Pop into stack

s = 0;

string[] dateVars;

for (int i = 0; i < input.Length; i++)

{

Date date = new Date();

if (input[i].Substring(0, 1).Equals("I"))

{

s = Convert.ToInt32(input[i].Substring(1, 2)) - 1;

dateVars = input[i].Split(new string[] { ",", " " }, StringSplitOptions.RemoveEmptyEntries);

for (int j = 1; j < dateVars.Length; j++)

{

int val = 0;

bool isNum = int.TryParse(dateVars[j], out val);

if (isNum)

if (val <= 31 && val >= 1)

date.day = val;

else

date.year = val;

else

date.month = dateVars[j];

}

// If push cannot be completed, even after any reallocation

// then stop processing, stack is full

if (!aDataStack.Push(s, date))

break;

}

else if (input[i].Substring(0, 1).Equals("D"))

{

s = Convert.ToInt32(input[i].Substring(1)) - 1;

aDataStack.Pop(s);

}

}

Console.Write("Final Stack Result for A Data: ");

cbDataStack.PrintStack();

Console.Write("\nPress any key to exit...");

Console.Read();

}

}

public class Date

{

public string month = "";

public int day = 0;

public int year = 0;

public override string ToString()

{

return month + " " + day + ", " + year;

}

}

}

**SuperStack.cs**

using System;

namespace MultiStack

{

class SuperStack<T>

{

int m = 0, n = 0, L0 = 0, lowBound = 0, upBound = 0, minSpace = 1;

T[] stack;

int[] sBase;

int[] sTop;

int[] oneArray; // combines growth, oltTop, and newBase Required for B option

// Creates an array in system stack to be used for Multi-Stack

public static SuperStack<T> CreateStack(int lowBound, int upBound, int l0, int size, int amtStacks)

{

// Generic SuperStack

SuperStack<T> ss = new SuperStack<T>();

// initialize all variables

ss.stack = new T[upBound - lowBound];

ss.sBase = new int[amtStacks + 1];

ss.sTop = new int[amtStacks];

ss.oneArray = new int[amtStacks + 1];

ss.minSpace = Floor(size \* 0.05f);

ss.lowBound = lowBound;

ss.upBound = upBound;

ss.m = size;

ss.L0 = l0;

ss.n = amtStacks;

// assign initial base stacks

for (int i = 0; i < amtStacks; i++)

{

ss.sBase[i] = ss.sTop[i] = ss.oneArray[i + 1] = (size - l0) / amtStacks \* i + l0;

}

// last (extra) base used to check overflow on last stack

ss.sBase[amtStacks] = size;

// return SuperStack to be used by main program

return ss;

}

public bool Push(int stackNum, T obj)

{

sTop[stackNum]++;

if (sTop[stackNum] > sBase[stackNum + 1])

{

Console.WriteLine("Stack overflow: Stack " + (stackNum+1) + " while pushing " + obj.ToString());

Console.WriteLine("Reallocating stack space...");

// If reallocate is a success

// then try to push the object into the stach again

// else stop processing

Console.Write("\nBefore Reallocate: ");

PrintStack();

bool success = Reallocate(0.15f);

Console.Write("\nAfter Reallocate: ");

PrintStack();

sTop[stackNum]--;

if (success)

Push(stackNum, obj);

else

return false;

}

else

{

stack[sTop[stackNum]] = obj;

Console.WriteLine("Pushing " + obj.ToString() + " to stack" + (stackNum + 1) + "[" + sTop[stackNum] + "]");

return true;

}

return true;

}

public T Pop(int stackNum)

{

T obj = default(T);

if (sTop[stackNum] == sBase[stackNum])

{

Console.WriteLine("Stack underflow: Stack " + (stackNum + 1)+ " is empty.");

return default(T);

}

else

{

obj = stack[sTop[stackNum]];

Console.WriteLine("Popping stack" + (stackNum + 1) + "[" + sTop[stackNum] + "] = \"" + obj + "\"");

stack[sTop[stackNum]] = default(T); // clear position to make printing cleaner

sTop[stackNum]--;

}

return obj;

}

bool Reallocate(float equalAllocate)

{

int availSpace = m - L0;

int totalInc = 0;

int j = n - 1;

for (int i = 0; i < n; i++)

{

availSpace -= sTop[i] - sBase[i];

totalInc += sTop[i] - oneArray[i + 1];

}

if (availSpace < minSpace)

{

Console.WriteLine("Insufficient memory for re-packing");

return false;

}

float growthAllocate = 1 - equalAllocate;

float alpha = equalAllocate \* availSpace / n;

float beta = growthAllocate \* availSpace / totalInc;

// oneArray being used as newBase

oneArray[0] = sBase[0];

float sigma = 0;

for (int i = 1; i <= n; i++)

{

// oneArray being used to calculate growth and the assigned newBase

float tau = sigma + alpha + Max(sTop[i - 1] - oneArray[i], 0) \* beta;

oneArray[i] = oneArray[i - 1] + (sTop[i - 1] - sBase[i - 1]) + Floor(tau) - Floor(sigma);

sigma = tau;

}

MoveStack();

for (int i = 0; i < n; i++)

{

// oneArray being used as oldTop after stacks have been moved

oneArray[i + 1] = sTop[i];

}

return true;

}

void MoveStack()

{

// try move stacks down

int delta = 0;

for (int i = 1; i < n; i++)

{

if (oneArray[i] < sBase[i])

{

delta = sBase[i] - oneArray[i];

for (int l = sBase[i]+1; l <= sTop[i]; l++)

{

stack[l - delta] = stack[l];

}

sBase[i] = oneArray[i];

sTop[i] -= delta;

}

}

// try move stacks up

delta = 0;

for (int i = n - 1; i >= 1; i--)

{

if (oneArray[i] > sBase[i])

{

delta = oneArray[i] - sBase[i];

for (int l = sTop[i]; l >= sBase[i]; l--)

{

stack[l + delta] = stack[l];

}

sBase[i] = oneArray[i];

sTop[i] += delta;

}

}

}

public void PrintStack()

{

Console.WriteLine();

for (int i = 0; i < n; i++)

{

Console.WriteLine("Stack " + i + " -- Base: " + sBase[i] + ", Top: " + sTop[i] + ", OldTop: " + oneArray[i + 1]);

}

int stackNum = 0;

for (int i = L0; i <= m; i++)

{

Console.WriteLine("Location " + i + " : Stack " + (stackNum + 1) + " : " + stack[i]);

if (i == sBase[stackNum + 1])

{

stackNum++;

}

}

Console.WriteLine();

}

static int Floor(float val)

{

int temp = (int)val;

if (temp <= val)

return temp;

else

return temp - 1;

}

static int Max(int a, int b)

{

if (a > b)

return a;

else

return b;

}

}

}

***inputCB.txt***

I1 Burris

I2 Zhou

I2 Shashidhar

I3 Shannon

I1 Yang

I3 Smith

D2

I1 Wei

I2 Rabieh

D1

D1

I2 Song

I2 Cho

D3

I2 Varol

I3 Karabiyik

I1 Cooper

I1 Smith

I1 McGuire

I3 Najar

I2 An

I1 Zhou

D2

I2 Deering

I1 Burris

I2 Cho

I2 McGuire

I3 Hope

I3 Pray

I3 NoHope

**inputA.txt**

I2 January 15, 1956

I2 February 14, 1957

I3 September 16, 1946

I2 September 17, 1842

I2 April 1, 2015

I1 December 24, 1996

D1

I3 March 16, 1992

D1

I2 January 15, 1956

I3 April 4, 1492

I3 November 7, 1776

I3 June 12, 1994

I2 July 4, 1776

I2 January 15, 2012

I3 December 6, 1991

I3 March 5, 1886

I1 October 24, 1996

I1 November 23, 1996

I1 November 2, 1990

I3 September 14, 1998

Enter a Lower Bound: -11

Enter a Upper Bound: 51

Enter the stack start index: 4

Enter the total amount of space for the stacks: 20

Enter the number of stacks: 4

Enter name of input file for C/B data: inputCB.txt

Pushing Burris to stack1[5]

Pushing Zhou to stack2[9]

Pushing Shashidhar to stack2[10]

Pushing Shannon to stack3[13]

Pushing Yang to stack1[6]

Pushing Smith to stack3[14]

Popping stack2[10] = "Shashidhar"

Pushing Wei to stack1[7]

Pushing Rabieh to stack2[10]

Popping stack1[7] = "Wei"

Popping stack1[6] = "Yang"

Pushing Song to stack2[11]

Pushing Cho to stack2[12]

Popping stack3[14] = "Smith"

Stack overflow: Stack 2 while pushing Varol

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 5, OldTop: 4

Stack 1 -- Base: 8, Top: 13, OldTop: 8

Stack 2 -- Base: 12, Top: 13, OldTop: 12

Stack 3 -- Base: 16, Top: 16, OldTop: 16

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 :

Location 7 : Stack 1 :

Location 8 : Stack 1 :

Location 9 : Stack 2 : Zhou

Location 10 : Stack 2 : Rabieh

Location 11 : Stack 2 : Song

Location 12 : Stack 2 : Cho

Location 13 : Stack 3 : Shannon

Location 14 : Stack 3 :

Location 15 : Stack 3 :

Location 16 : Stack 3 :

Location 17 : Stack 4 :

Location 18 : Stack 4 :

Location 19 : Stack 4 :

Location 20 : Stack 4 :

After Reallocate:

Stack 0 -- Base: 4, Top: 5, OldTop: 5

Stack 1 -- Base: 6, Top: 11, OldTop: 11

Stack 2 -- Base: 17, Top: 18, OldTop: 18

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 :

Location 7 : Stack 2 : Zhou

Location 8 : Stack 2 : Rabieh

Location 9 : Stack 2 : Song

Location 10 : Stack 2 : Cho

Location 11 : Stack 2 : Shannon

Location 12 : Stack 2 : Cho

Location 13 : Stack 2 : Shannon

Location 14 : Stack 2 :

Location 15 : Stack 2 :

Location 16 : Stack 2 :

Location 17 : Stack 2 : Cho

Location 18 : Stack 3 : Shannon

Location 19 : Stack 3 :

Location 20 : Stack 4 :

Pushing Varol to stack2[11]

Pushing Karabiyik to stack3[19]

Pushing Cooper to stack1[6]

Stack overflow: Stack 1 while pushing Smith

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 7, OldTop: 5

Stack 1 -- Base: 6, Top: 11, OldTop: 11

Stack 2 -- Base: 17, Top: 19, OldTop: 18

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 2 : Zhou

Location 8 : Stack 2 : Rabieh

Location 9 : Stack 2 : Song

Location 10 : Stack 2 : Cho

Location 11 : Stack 2 : Varol

Location 12 : Stack 2 : Cho

Location 13 : Stack 2 : Shannon

Location 14 : Stack 2 :

Location 15 : Stack 2 :

Location 16 : Stack 2 :

Location 17 : Stack 2 : Cho

Location 18 : Stack 3 : Shannon

Location 19 : Stack 3 : Karabiyik

Location 20 : Stack 4 :

After Reallocate:

Stack 0 -- Base: 4, Top: 7, OldTop: 7

Stack 1 -- Base: 10, Top: 15, OldTop: 15

Stack 2 -- Base: 15, Top: 17, OldTop: 17

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Zhou

Location 8 : Stack 1 : Rabieh

Location 9 : Stack 1 : Song

Location 10 : Stack 1 : Cooper

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 3 : Shannon

Location 17 : Stack 3 : Karabiyik

Location 18 : Stack 3 : Shannon

Location 19 : Stack 3 : Karabiyik

Location 20 : Stack 4 :

Pushing Smith to stack1[7]

Pushing McGuire to stack1[8]

Pushing Najar to stack3[18]

Stack overflow: Stack 2 while pushing An

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 8, OldTop: 7

Stack 1 -- Base: 10, Top: 16, OldTop: 15

Stack 2 -- Base: 15, Top: 18, OldTop: 17

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Song

Location 10 : Stack 1 : Cooper

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 3 : Shannon

Location 17 : Stack 3 : Karabiyik

Location 18 : Stack 3 : Najar

Location 19 : Stack 3 : Karabiyik

Location 20 : Stack 4 :

After Reallocate:

Stack 0 -- Base: 4, Top: 8, OldTop: 8

Stack 1 -- Base: 8, Top: 14, OldTop: 14

Stack 2 -- Base: 15, Top: 18, OldTop: 18

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 2 : Zhou

Location 10 : Stack 2 : Rabieh

Location 11 : Stack 2 : Song

Location 12 : Stack 2 : Cho

Location 13 : Stack 2 : Varol

Location 14 : Stack 2 : Shannon

Location 15 : Stack 2 : Varol

Location 16 : Stack 3 : Shannon

Location 17 : Stack 3 : Karabiyik

Location 18 : Stack 3 : Najar

Location 19 : Stack 3 : Karabiyik

Location 20 : Stack 4 :

Pushing An to stack2[14]

Stack overflow: Stack 1 while pushing Zhou

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 9, OldTop: 8

Stack 1 -- Base: 8, Top: 14, OldTop: 14

Stack 2 -- Base: 15, Top: 18, OldTop: 18

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 2 : Zhou

Location 10 : Stack 2 : Rabieh

Location 11 : Stack 2 : Song

Location 12 : Stack 2 : Cho

Location 13 : Stack 2 : Varol

Location 14 : Stack 2 : An

Location 15 : Stack 2 : Varol

Location 16 : Stack 3 : Shannon

Location 17 : Stack 3 : Karabiyik

Location 18 : Stack 3 : Najar

Location 19 : Stack 3 : Karabiyik

Location 20 : Stack 4 :

After Reallocate:

Stack 0 -- Base: 4, Top: 9, OldTop: 9

Stack 1 -- Base: 10, Top: 16, OldTop: 16

Stack 2 -- Base: 16, Top: 19, OldTop: 19

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Zhou

Location 10 : Stack 1 : McGuire

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 2 : An

Location 17 : Stack 3 : Shannon

Location 18 : Stack 3 : Karabiyik

Location 19 : Stack 3 : Najar

Location 20 : Stack 4 :

Pushing Zhou to stack1[9]

Popping stack2[16] = "An"

Pushing Deering to stack2[16]

Pushing Burris to stack1[10]

Stack overflow: Stack 2 while pushing Cho

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 10, OldTop: 9

Stack 1 -- Base: 10, Top: 17, OldTop: 16

Stack 2 -- Base: 16, Top: 19, OldTop: 19

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Zhou

Location 10 : Stack 1 : Burris

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 2 : Deering

Location 17 : Stack 3 : Shannon

Location 18 : Stack 3 : Karabiyik

Location 19 : Stack 3 : Najar

Location 20 : Stack 4 :

Insufficient memory for re-packing

After Reallocate:

Stack 0 -- Base: 4, Top: 10, OldTop: 9

Stack 1 -- Base: 10, Top: 17, OldTop: 16

Stack 2 -- Base: 16, Top: 19, OldTop: 19

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Zhou

Location 10 : Stack 1 : Burris

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 2 : Deering

Location 17 : Stack 3 : Shannon

Location 18 : Stack 3 : Karabiyik

Location 19 : Stack 3 : Najar

Location 20 : Stack 4 :

Final Stack Result for C/B Data:

Stack 0 -- Base: 4, Top: 10, OldTop: 9

Stack 1 -- Base: 10, Top: 16, OldTop: 16

Stack 2 -- Base: 16, Top: 19, OldTop: 19

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Zhou

Location 10 : Stack 1 : Burris

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 2 : Deering

Location 17 : Stack 3 : Shannon

Location 18 : Stack 3 : Karabiyik

Location 19 : Stack 3 : Najar

Location 20 : Stack 4 :

Enter a Lower Bound: 0

Enter a Upper Bound: 50

Enter the stack start index: 4

Enter the total amount of space for the stacks: 13

Enter the number of stacks: 3

Enter name of input file for A data: inputA.txt

Pushing January 15, 1956 to stack2[8]

Pushing February 14, 1957 to stack2[9]

Pushing September 16, 1946 to stack3[11]

Pushing September 17, 1842 to stack2[10]

Stack overflow: Stack 2 while pushing April 1, 2015

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 7, Top: 11, OldTop: 7

Stack 2 -- Base: 10, Top: 11, OldTop: 10

Location 4 : Stack 1 :

Location 5 : Stack 1 :

Location 6 : Stack 1 :

Location 7 : Stack 1 :

Location 8 : Stack 2 : January 15, 1956

Location 9 : Stack 2 : February 14, 1957

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 3 : September 16, 1946

Location 12 : Stack 3 :

Location 13 : Stack 3 :

After Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 8, OldTop: 8

Stack 2 -- Base: 11, Top: 12, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : September 16, 1946

Location 9 : Stack 2 : February 14, 1957

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 2 : September 17, 1842

Location 12 : Stack 3 : September 16, 1946

Location 13 : Stack 3 :

Pushing April 1, 2015 to stack2[8]

Stack overflow: Stack 1 while pushing December 24, 1996

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 5, OldTop: 4

Stack 1 -- Base: 4, Top: 8, OldTop: 8

Stack 2 -- Base: 11, Top: 12, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : February 14, 1957

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 2 : September 17, 1842

Location 12 : Stack 3 : September 16, 1946

Location 13 : Stack 3 :

After Reallocate:

Stack 0 -- Base: 4, Top: 5, OldTop: 5

Stack 1 -- Base: 7, Top: 11, OldTop: 11

Stack 2 -- Base: 11, Top: 12, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 1 : January 15, 1956

Location 6 : Stack 1 : February 14, 1957

Location 7 : Stack 1 :

Location 8 : Stack 2 : January 15, 1956

Location 9 : Stack 2 : February 14, 1957

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 2 : April 1, 2015

Location 12 : Stack 3 : September 16, 1946

Location 13 : Stack 3 :

Pushing December 24, 1996 to stack1[5]

Popping stack1[5] = "December 24, 1996"

Pushing March 16, 1992 to stack3[13]

Stack underflow: Stack 1 is empty.

Stack overflow: Stack 2 while pushing January 15, 1956

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 5

Stack 1 -- Base: 7, Top: 12, OldTop: 11

Stack 2 -- Base: 11, Top: 13, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 1 :

Location 6 : Stack 1 : February 14, 1957

Location 7 : Stack 1 :

Location 8 : Stack 2 : January 15, 1956

Location 9 : Stack 2 : February 14, 1957

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 2 : April 1, 2015

Location 12 : Stack 3 : September 16, 1946

Location 13 : Stack 3 : March 16, 1992

After Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 9, OldTop: 9

Stack 2 -- Base: 10, Top: 12, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : September 16, 1946

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 3 : September 16, 1946

Location 12 : Stack 3 : March 16, 1992

Location 13 : Stack 3 : March 16, 1992

Pushing January 15, 1956 to stack2[9]

Pushing April 4, 1492 to stack3[13]

Stack overflow: Stack 3 while pushing November 7, 1776

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 9, OldTop: 9

Stack 2 -- Base: 10, Top: 14, OldTop: 12

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : January 15, 1956

Location 10 : Stack 2 : September 17, 1842

Location 11 : Stack 3 : September 16, 1946

Location 12 : Stack 3 : March 16, 1992

Location 13 : Stack 3 : April 4, 1492

After Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 9, OldTop: 9

Stack 2 -- Base: 9, Top: 13, OldTop: 13

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : January 15, 1956

Location 10 : Stack 3 : September 16, 1946

Location 11 : Stack 3 : March 16, 1992

Location 12 : Stack 3 : April 4, 1492

Location 13 : Stack 3 :

Pushing November 7, 1776 to stack3[13]

Stack overflow: Stack 3 while pushing June 12, 1994

Reallocating stack space...

Before Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 9, OldTop: 9

Stack 2 -- Base: 9, Top: 14, OldTop: 13

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : January 15, 1956

Location 10 : Stack 3 : September 16, 1946

Location 11 : Stack 3 : March 16, 1992

Location 12 : Stack 3 : April 4, 1492

Location 13 : Stack 3 : November 7, 1776

Insufficient memory for re-packing

After Reallocate:

Stack 0 -- Base: 4, Top: 4, OldTop: 4

Stack 1 -- Base: 4, Top: 9, OldTop: 9

Stack 2 -- Base: 9, Top: 14, OldTop: 13

Location 4 : Stack 1 :

Location 5 : Stack 2 : January 15, 1956

Location 6 : Stack 2 : February 14, 1957

Location 7 : Stack 2 : September 17, 1842

Location 8 : Stack 2 : April 1, 2015

Location 9 : Stack 2 : January 15, 1956

Location 10 : Stack 3 : September 16, 1946

Location 11 : Stack 3 : March 16, 1992

Location 12 : Stack 3 : April 4, 1492

Location 13 : Stack 3 : November 7, 1776

Final Stack Result for A Data:

Stack 0 -- Base: 4, Top: 10, OldTop: 9

Stack 1 -- Base: 10, Top: 16, OldTop: 16

Stack 2 -- Base: 16, Top: 19, OldTop: 19

Stack 3 -- Base: 19, Top: 19, OldTop: 19

Location 4 : Stack 1 :

Location 5 : Stack 1 : Burris

Location 6 : Stack 1 : Cooper

Location 7 : Stack 1 : Smith

Location 8 : Stack 1 : McGuire

Location 9 : Stack 1 : Zhou

Location 10 : Stack 1 : Burris

Location 11 : Stack 2 : Zhou

Location 12 : Stack 2 : Rabieh

Location 13 : Stack 2 : Song

Location 14 : Stack 2 : Cho

Location 15 : Stack 2 : Varol

Location 16 : Stack 2 : Deering

Location 17 : Stack 3 : Shannon

Location 18 : Stack 3 : Karabiyik

Location 19 : Stack 3 : Najar

Location 20 : Stack 4 :

Press any key to exit...