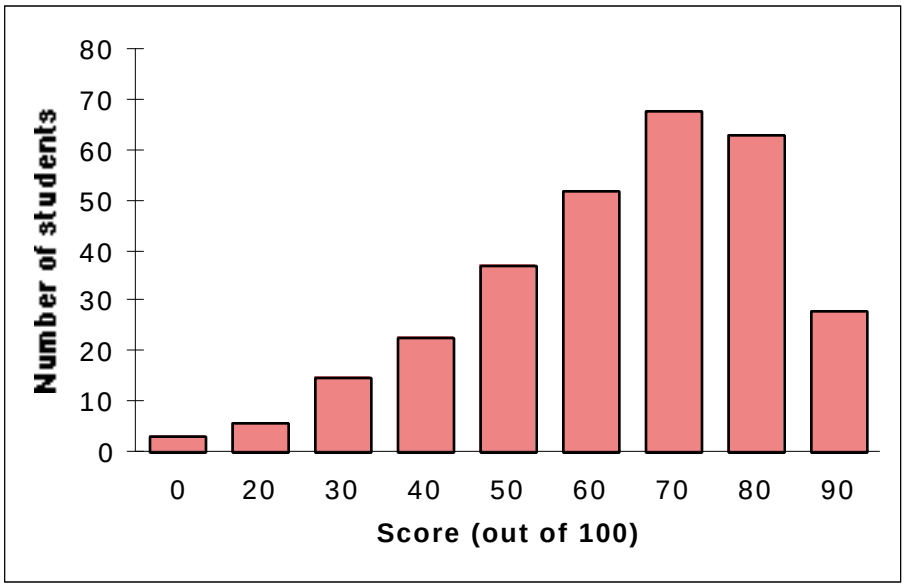
CS106A
Winter 99-00

Handout #42
March 17, 2000

Final exam solution

Exam stats: median 74, mean 70 and standard deviation 18. Full histogram below:



I tired to make the final a bit more intense than the midterm, to help give a little more room in the curve and that seemed to be somewhat successful. It was by no means an easy exam, but it appeared that almost everyone had a chance to make a decent attempt at all the questions. Most folks coming out of the exam seemed to be in good spirits—glad to be done and looking forward to enjoying the beautiful weather!

Your course grade is comprised of 45% homework, 10% section participation, 15% midterm and 30% final. For the students whose final showed improvement over their midterm, I counted the final a bit more and the midterm a bit less. As promised, I was pretty generous with the A and B grades for the course. In our class of 310 students, the course grades were 151 A’s, 132 B’s, 19 C’s and 5 D’s.

True to CS106A's reputation, those grades were earned only with a big investment of your time and hard work. I hope many of you will join in 106B for even more interesting adventures in the world of computer programming!

Problem 1: Short answer (16 points)

a) SelectionSort on the left, InsertionSort on the right.

b) During execution, it will write off the end of the allocated space when trying to concatenate the second string onto the first.

c)

```
pointT CreatePoint(int x, int y)
{
   pointT p;

   p = New(pointT);
   p->x = x;
   p->y = y;
   return p;
}
```

d)

```
grid[row][col] = (row + col) % GridSize;
```

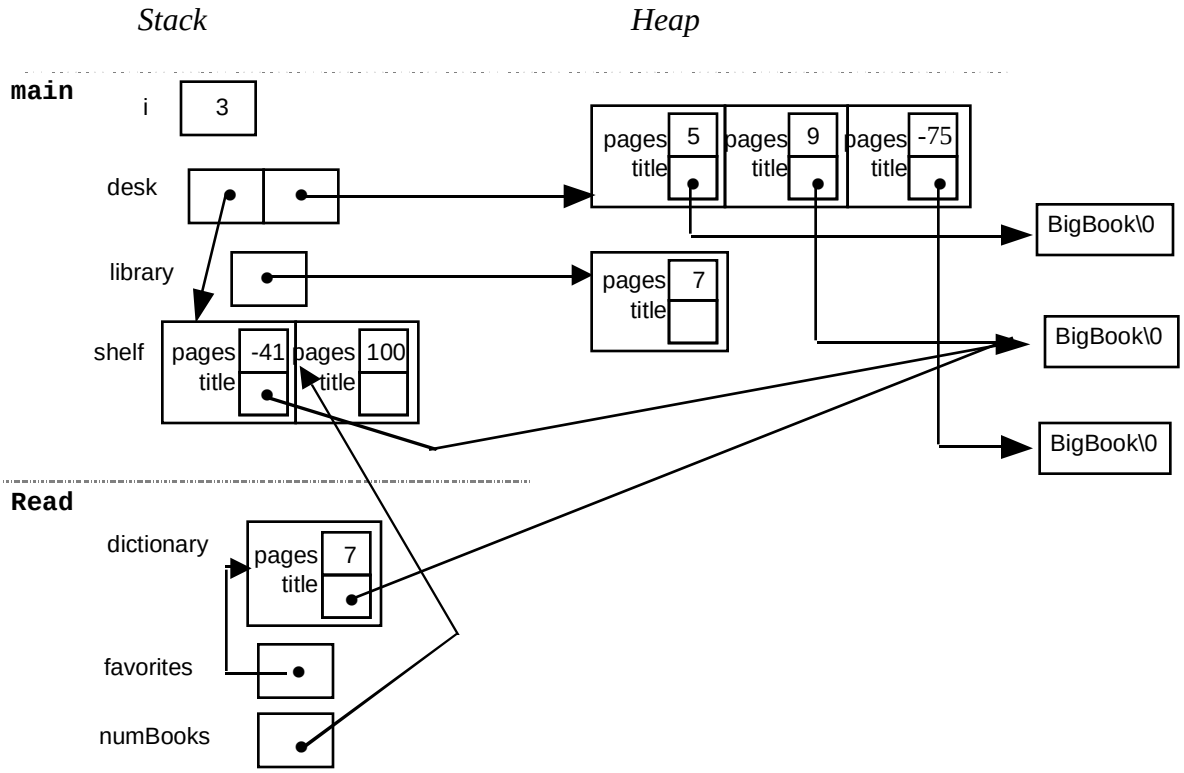
e) When the element you are searching for is duplicated in the array, Linear Search will always find and return the first occurrence. Binary Search will return the index of the element first found at a midpoint, which may be the first, second, last, whatever of the duplicates.

f) The min and max parameters are ordinary integers and thus passed by value, meaning a copy is made and only the copy is changed. We need to get that information back out of the function and thus need to pass pointers to the two integers to perform call by reference. This means changing the prototype to declare both min and max as int*s, change the usage in the function to include a deference, and pass the address of min and max from main rather than the values.

Problem 2: File processing (22 points)

```
#define MaxCand 20

typedef struct {
   string name;
   int nVotes;
} candidateT;


/* returns whether new candidate was added or not */
static bool TallyOne(string name, candidateT cand[], int nCand)
{
   int i;

   for (i = 0; i < nCand; i++) {
      if (StringEqual(cand[i].name, name)) { /* if found in array */
         cand[i].nVotes++;                   /* add one more vote */
         return FALSE;
      }
   }
   cand[nCand].name = name;   /* not found, add name to end of array */
   cand[nCand].nVotes = 1;    /* start with this 1 vote */
   return TRUE;
}


static void TallyVotes(FILE *in)
{
   candidateT cand[MaxCand];
   int nCand = 0;
   string name;
   int i, totalVotes = 0;

   while ((name = ReadLine(in)) != NULL) {
      if (TallyOne(name, cand, nCand))
          nCand++;        /* increment nCand if new name added */
      totalVotes++;
   }
   for (i = 0; i < nCand; i++)
      printf("%s %d%%\n", cand[i].name,(cand[i].nVotes*100)/totalVotes);
}
```

Problem 3: Pointers and program tracing (20 points)



Problem 4: Data structures (32 points)

4a) Data structure access (6 points)

```
classT *

studentT

string *

univ.best[Math]->nEnrolled

univ.student[5].sl.nClasses

univ.student[0].sl.schedule[0]->dept
```

4b) The `Enroll` function (10 points)

```
static bool Enroll(universityT *u, int s, classT *cp)
{
   if (u->student[s].sl.nUnits + cp->nUnits > MaxUnits
         || cp->nEnrolled >= cp->capacity)
           return FALSE;
   u->student[s].sl.schedule[u->student[s].sl.nClasses++] = cp;
   u->student[s].sl.nUnits += cp->nUnits;
   cp->nEnrolled++;
   return TRUE;
}
```

4c) The `ComputeBest` function (16 points)

```
static void ComputeBest(universityT *u)
{
    int i, dept, minSlack, slack;

    for (dept = Art; dept <= Psych; dept++) {
    minSlack = -1;
    for (i = 0; i < u->nClasses; i++) {
        if (u->class[i].dept == dept) {
           slack = u->class[i].capacity - u->class[i].nEnrolled;
           if (minSlack == -1 || slack < minSlack) {
              minSlack = slack;
              u->best[dept] = &u->class[i];
           }
        }
    }
}
```

Problem 5: Strings Revealed (10 points)

```
string Acronym(string words[], int n)
{
    string result;
    int i;

    result = GetBlock(n + 1); /* one more for null terminator */
    for (i = 0; i < n; i++)
       result[i] = toupper(words[i][0]);
    result[n] = '\0';
    return result;
}
```