CS 101
Computer Programming and utilization

IIT BOMBAY

Dr Deepak B Phatak
Subrao Nilekani Chair Professor
Department of CSE, Kanwal Rekhi Building
IIT Bombay

Session 14, Analysis of midsem questions

Tuesday and Wednesday, September 21 and 22, 2010

Q1 (a)

- Given a number m which has odd digits, find and print its middle digit d

```
 /* alternate solution I */
/* find number of digits in m */
temp1 = m;   i = 1;
while (temp1 > 9){
   temp1 = temp1/10;
   i++;
}
cout << "Number of digits is " << i << endl;
if (i%2 == 0){ cout << " even digits ivalid" << endl;
                      return 1;};
```

Sample program for Q1(a) - Continued

```
/* To preserve m, we assign it to temp1 and use that variable
  The middle digit is at position i/2 +1
  To get that, we first evaluate temp2 = 10 raised to power (i/2)
   Then we divide temp1 by temp2 to get the middle digit in the
    last position, and extract it using modulo 10
 */
 temp2 = 1;
 for (j = 0; j < i/2; j++){
    temp2 = temp2 * 10;
 }
 temp1 = m / temp2;
 k = temp1%10;
 cout << "Middle digit is: " << k << endl;
```

Alternate solution to Q 1(a)

```
temp1=m;
  k=1;
  k=temp1%10;
  temp2=temp1/10;
  while(temp1/100!=0){
    k=temp2%10; temp2=temp2/10;
    temp1=temp1/100;
  }
  cout<<"the middle digit of the number"<< m <<" is " <<k<<endl;
  return 0;
}
```

Q 1 (b)

- Given two real numbers x and y, find and print the value of z using following logic:

if x < 0 and if y > 0 then z = y-x; if x < 0 and y <=0 then z = -y-x;

if x >=0 and if y > 0 then z= x+y; if x >=0 and y <=0 then z = x-y;

(The given logic adds absolute values of given numbers)

```
 /*  Q1 b */
cout << "Give values of x and y: " << endl;
cin >> x >> y;
```

Sample solution to Q 1 (b)

```
/* alternnate soluion I */
  if (x < 0) {
    if (y > 0) {
      z = y-x;
    }
    else {
      z = -y-x;
    }
  }
```

Q1(b) ...

```
else {
   if(y > 0){
      z = x+y;
   }
   else{
      z = x-y;
   }
}
cout << "Q1 (b), Solution I; Value of z is: " << z << endl;
```

Q1(b) …

```
/* alternate solution II */

    if (x < 0) if (y > 0)z = y-x;
                    else z= -y-x;
   else if(y>0) z= x+y;
           else  z = x-y;
   cout << "Q1 (b), Solution II; Value of z is: " << z << endl;
```

Q1 (b) …

```
/* alternate solution III */
if (x < 0 && y > 0)z = y-x;
if (x < 0 && y <= 0)z = -y-x;
if (x >= 0 && y > 0)z = x+y;
if (x >= 0 && y <= 0)z = x-y;
cout << "Q1 (b), Solution III; Value of z is: " << z << endl;
```

Q1(b) …

```
/* alternnate soluion IV */
/*  We note that the given logic actually finds the sum of abosolute
    values of x and y. That is:
   z = |x| + |y|
   We can use the standard library function abs()
   This solution requires use of standard maths library
   We must include the following at the beginning of the program
    #include <cmath>
*/
  z = abs(x) + abs  (y);
   cout << "Q1 (b) Soltion IV; Value of z is: " << z << endl;
```

Sample solution to Q1 (c)

```
 /*  Q1 c*/
//  Given n elements of the array a[],
//   find the second largest element max2
   cout << "give the number of elements " << endl;
   cin >> n;
   if (n<2 || n > 10) { cout <<"Invalid number"<<endl; return 1;}
   cout << "Give values of " << n << " elements" << endl;
   for (i=0; i<n; i++) cin >> a[i];
   //  We assume values to be [ 5  7  24  18  72  71]
   /* alternate solution I *   /
```

Q1 (c) …

```
//  Assumed values [ 5  7  24  18  72  71]
/* We use temp1 to store the largest, and temp2 to store the
   second largest value at any stage of our array scan.
   We start with the assumption that the 0th and 1st elements
   are largest and second largest. If these are not in that order,
   we swap these */
   temp1 = a[0]; temp2 = a[1];
   if (a[1] > a[0]){
      temp1 = a[1];
      temp2 = a[0];
   }
```

Q1 (c) …

```
/* We examine the array from 2nd element onward. If that element
   is larger than temp1, we move temp1 to temp2   and assign that
   element (now the largest) to temp1.  If that is not so, we compare
   it with temp2,  and change temp2 if necessary.
   At the end of examining all elements of a[], we will have temp2
   holding the second maximum value */
  //                                      Assumed values [ 5  7  24  18  72  71]
   for (i = 2; i < n; i++){
      if (a[i]  > temp1){
         temp2 = temp1;  temp1 = a[i];
      }
      else if (a[i] > temp2) temp2 = a[i];
   }
   cout << "Second maximum element is: " << temp2 << endl;
```

Q1 (c) ...

```
cout<<"give the value of n between 1 to 10"<<endl;  cin>>n;
 cout<<"enter the elements"<<endl;  for(i=0; i<n; i++)  cin>> a[i];
 /* Alternate Solution II  */
 d=a[0];                    //  Assumed values [ 5  7  24  18  72  71]
 for(i=1;i<n;i++){
   if (d < a[i]) {
     max2 = d;
     d = a[i];
   }
 }
 cout<<"the second maximum in the array is " <<max2<<endl;
```

Q 1 (c) …

```
/* Alternate Solution III */
// The array is to be sorted in descending order, a[0] will be largest,
// so we pick up the next element a[1]
…..
max2 = a[1]
cout<<"the second maximum in the array is " <<max2<<endl;
//  Assumed values [ 5  7  24  18  72  71]
```

Q1 (d)

```
/* Given a positive integer q between 1 and 39, and a character c,
   prepare a null terminated string in the array str[] by filling up c in
   the first q positions of the array.*/
   cout << "give a character"  << endl;
   cin >> c;
   cout << endl<< "how many times to be inserted in the string? ";
   cout << endl;
   cin >> q;
   if (q <=0 || q > 38) { cout << "invalid number" << endl; return 1;}
```

Q1 (d) …

```
/* alternnate soluion I */

 /* fill up the string */
   for (i = 0; i < q ; i ++) str[i] = c;
   str[q] = '\0';
   cout << " String is: " << str << endl;
```

Q1 (d) …

```
 /* alternate soluion II */

 for(i=0;i<q-1;i++)  str[i]=c;
 str[i] ='\0';

cout<<"printing the characters"<<endl;
for(i=0;i<q;i++)  cout<<str[i]<<" ";
cout<<endl;
cout<<str<<endl;
```

Sample solution to Q 2

```
#include <iostream>
using namespace std;
/* Sample solution to Q 2 (a)*/
int calculate_check_digit ( int N ) {
    /* This function calculates check digit for a given number */
    int sum =0, i, d, cd;   // cd will have calculated check digit
    while (N >0){
       d = N%10;  N = N/10;  sum = sum + d;
    }
    cd = sum %10;
    return cd;
}
```

Q2 (b) – main program

```
int main(){
   int EncodedNumber, CheckDigit, OriginalNumber , cd;
   cout << "Give a encoded number: ";   cin >> EncodedNumber;
   if (EncodedNumber <=9) {
      cout << endl << "Invalid encoded number" << endl;
      return 1;
   }
```

Q2 (b) ...

```
  CheckDigit = EncodedNumber%10;  // extract the check digit
  cout << "Checkdigit of the Given number is: " ;
  cout  << CheckDigit << endl;
  OriginalNumber = EncodedNumber/10 ;  //get the original number
  cd = calculate_check_digit(OriginalNumber);
  // use function to calulate what should be the correct check digit

  if (cd == CheckDigit) {cout << " Given number is valid" << endl;}
  else {  cout <<"check digit should be " << cd ;
              cout <<  " Invalid    number"<<endl; }
 return 0;
}
```

Sample solution to Q3

```
/* progQ3.cpp
    This is a modified version of the program top_performers.cpp
    discussed in the class. It generates artificial data for marks for up
    to 1 Lakh students by using a sequence from 0 to N-1. Alternate
    values are put into top and bottom halves of the array
    respectively.

    The program is required to sort the marks in descending order,
    the top half and bottom halves of the marks array are to be sorted
    independently, but within the same array. It is then required to
    merge the two parts into the result array topmarks[] */
```

Q3 …

```
#include <iostream>
using namespace std;
int main(){
   int marks[100000], max, N, i, j, k, pos, temp;
   int topmarks[100000];
   cout << "Give number of students "; cin >> N;
   if (N<=0 || N > 100000 || N%2 !=0){cout <<"Invalid N\n" ; return 1;}
   cout << "generating artificial marks data for " << N;
   cout << " students" << endl;
   for (i=0, j=N/2; i < N/2; i++, j++) {
      marks[i] = 2*i+1; marks[j] = 2*i+2;
   }
```

Q3 …

```
 cout << "Proceeding to sort first part of the array" << endl;
 /* Your answer to part (a) should fit here */

for (k=0;  k<N/2 ; k++){
    max = marks[k]; pos =k;
    for (i=k+1; i < N/2; i++) {
       if (marks[i] > max) {  max = marks[i];  pos = i; }
    }
    temp = marks[k]; marks[k] = marks[pos]; marks[pos] = temp;
 }
```

Q3 …

```
cout << "Proceeding to sort second part of the array" << endl;
/* Your answer to part (b) should fit here */

/* sort second part of the array */
  for (k=N/2;  k<N ; k++){
   max = marks[k]; pos =k;
   for (i=k+1; i < N; i++) {
      if (marks[i] > max) {  max = marks[i];  pos = i; }
   }
   temp = marks[k]; marks[k] = marks[pos]; marks[pos] = temp;
}
```

Q3 …

```
  cout << "Top 5 marks in first part are: " ;
  for (i=0; i<5; i++) cout <<marks[i] << ", ";
  cout << endl;"Top 5 marks from second part are: ";
  for (i=N/2; i<N/2+5; i++) cout <<marks[i]<< ", ";



 /* now merge the two parts of arrays into toproll*/
     /* Your answer to part (c) should fit here */
// sentinel insertion in the array parts not used as space is not
// available within the array. Instead, we use a separate variable
// to provide a sentinel value
```

Q3 …

```
int p=0, q=N/2; i=0;  int s = -999;  int val1, val2;
while (i < N){
    if (p < N/2) val1 = marks[p];else val1 =s;
    if (q < N) val2 = marks[q];else val2 =s;
    if (val1 > val2){
       topmarks[i]=marks[p];  p++;
    }
    else {
       topmarks[i]=marks[q]; q++;
    }
    i++;
}
```

Q3 …

```
    cout << endl << "Top 5 marks from combined list are: ";
    for (i=0; i<5; i++) cout  <<topmarks[i] << ", ";
    cout << endl;
    return 0;
}
```

Sample solution to Q4

```
/* progQ4.cpp
Sample answer to midsem exam Q4.
Applying median filter to a digital image
as per logic specified in the question
*/
#include<iostream>
using namespace std;
```

Q4 …

```
int main(){
  /* part (a) */
  short int image[1200][1200], newimage[1200][1200]; // pixel arrays
  int h, w, i, j, k, m, i1,j1; //other variables
  int swi, swj, neb[100],ws=3, wsize=9, pmax, ppos, ptemp, pmed;
  // window to hold neighbouring pixels, larger window is possible
  // Your answer in exam could simply use window of size 9
```

Q4 …

```
/* reading image matrix. The image data is expected to be
   prepared in a text file in the following format:
first line: contains two numbers: height (h) and width (w)
subsequent lines: contain pixel values,  w pixels for each of h rows
sample data for a 8 by 8 image kept in a file imagagedata.txt  */
cin >> h >> w; // read height and width of the given image
if (w < 3 || w > 1200 || h < 3 || w > 1200){
   /* checking validity of window size,
   an image smaller than 3 pixels is assumed to be wrong*/
   cout << "Image size inappropriate for filtering" << endl;
   return 1;
}
```

Q4 …

```
int inputerrorflag=0; // checking and reporting health for each pixel
for(i=0;i< h;i++){
  for(j=0;j < w;j++){
    cin >> image[i][j]; cout << image[i][j] << " " ;
    if (image[i][j] <0 || image[i][j] > 255){
      cout<< endl  << i << ", " << j <<"  " <<image[i][j]<<endl;
      inputerrorflag ++;
    }
  } cout << endl;
}
```

Q4 …

```
/* Quit if any error, continue only if all pixels are OK */
if (inputerrorflag != 0){
   cout << "Total erroneous pixels found: " << inputerrorflag << 
  endl;
   cout << "please correct the input file and rerun the program" << 
  endl;
   return 2;
}
// image is OK, proceed to filtering
```

Q4 …

```
/* part (b) (c) and (d) - Sliding window for all the inner pixels */
 for (i=0; i< h; i++){
   for (j=0; j < w; j++){    // pixel at i, j is the central pixel
     m = 0;
     for (m = 0; m < wsize;  m++){ // collect wsize pixels from window
       i1 = i+m/ws-1; j1 = j+m%ws-1;// ensuring right index values
```

Explanatory slide (Q4)

```
for (m = 0; m < wsize;  m++){ // collect wsize pixels from window
      i1 = i+m/ws-1; j1 = j+m%ws-1;// ensuring right index values
```

Q4 …

```
    if (i1 < 0) i1++; // ensuring right index for pixels on edges
    if (j1 <0) j1++;
    if (i1 ==h) i1--;
    if (j1 == w) j1--;
    neb[m] = image[i1][j1];
  }
```

Q4 …

```
    // sort the array of pixels in window
    for(i1=0; i1< wsize; i1++){
      for (j1 = i1+1; j1 < wsize; j1++){
        if (neb[j1] > neb[i1]){
          ptemp = neb[i1]; neb[i1]=neb[j1]; neb[j1]=ptemp;
        }
      }
    }
    newimage[i][j] = neb[wsize/2];// assign median to result element
  }
}
```

Q4 …

```
 /*  part (e) Print results in the same format as input image */
 cout << h << " " << w << endl;
 for (i=0; i<h; i++){
    for (j = 0; j < w; j++){
        cout << newimage[i][j] << " ";
    }
  cout << endl;
  }
  return 0;
}
```