PROBLEMS

# 1.

Read from standard input data for unknown number of students (not more then 100). Each row of the data is in following format:

• first name

• last name

• ID number (format xxyzzzz)

• four numbers (points for each problem) separated with tab space.

Write a program that will print list of students, where each row will have: last name, first name, number, and total points sorted by the number of points. BTW the names should be printed with first capital letter.

# 2.

Write a program that will read from standard input data for countries and will print on the standard output the name of the president of the country whose capital has largest population.

• Data for country: name, president, capital and population.

• Data for city: name and population.

• Data for president: name, political party

SOLUTIONS

# 1.

#include <iostream>  
#include <cctype>  
#include <cstring>  
*using namespace* std;  
  
*struct* student{  
 *char* name[20];  
 *char* surname[20];  
 *char* index[6];  
 *int* points[4];  
 *void* readStudent(){  
 **cin**>>name>>surname>>index;  
 *for*(*int* **i**=0 ; **i**<4 ; **i**++){  
 **cin**>>points[**i**];  
 }  
 }  
 *void* Normalize(*char* \*str){  
 str[0]= toupper(str[0]);  
 *for*(*int* **i**=1 ; **i**< strlen(str) ; **i**++){  
 str[**i**]= tolower(str[**i**]);  
 }  
 }  
 *int* totalPoints(){  
 *int* **sum**=0;  
 *for*(*int* **i**=0 ; **i**<4 ; **i**++){  
 **sum**+=points[**i**];  
 }  
 *return* **sum**;  
 }  
 *int* grade(){  
 *int* **p**=totalPoints();  
 *int* **gr**=**p**/10+1;  
 *if*(**gr**<5){  
 **gr**=5;  
 }  
 *if*(**gr**>10){  
 **gr**=10;  
 }  
 *return* **gr**;  
 }  
 *void* printStudent(){  
 Normalize(name);  
 Normalize(surname);  
 *int* **p**=totalPoints();  
 *int* **g**=grade();  
 **cout**<<name<<" "<<surname<<" "<<index<<" "<<**p**<<" "<<**g**<<endl;  
 }  
 *int* studentCmp(student other){  
 *return* grade()<=other.grade();  
 }  
};  
*void* swap(student &s1, student &s2){  
 student **tmp**=s1;  
 s1=s2;  
 s2=**tmp**;  
}  
*void* sort(student \*s, *int* n){  
 *for*(*int* **i**=0 ; **i**<n ; **i**++){  
 *for*(*int* **j**=0 ; **j**<n-1-**i** ; **j**++){  
 *if*(s[**j**].studentCmp(s[**j**+1])){  
 //student tmp=s[j];  
 //s[j]=s[j+1];  
 //s[j+1]=tmp;  
 swap(s[**j**],s[**j**+1]);  
 }  
 }  
 }  
}  
  
*int* main(){  
 *int* **n**;  
 **cin**>>**n**;  
 student **students**[100];  
 *for*(*int* **i**=0 ; **i**<**n** ; **i**++){  
 **students**[**i**].readStudent();  
 }  
 sort(**students**,**n**);  
 *for*(*int* **i**=0 ; **i**<**n** ; **i**++){  
 **students**[**i**].printStudent();  
 }  
 *return* 0;  
}

# 2.

#include <iostream>  
#include <cctype>  
#include <cstring>  
*using namespace* std;  
  
*struct* politician{  
 *char* pName[20];  
 *char* party[20];  
 *void* readPolitician(){  
 **cin** >> pName >> party;  
 }  
 *void* printPolitician(){  
 **cout** << pName << " " << party << " ";  
 }  
};  
*struct* city{  
 *char* cName[20];  
 *int* population;  
 *void* readCity(){  
 **cin** >> cName >> population;  
 }  
 *void* printCity(){  
 **cout** << cName << " " << population << " ";  
 }  
};  
  
*struct* country{  
 *char* name[20];  
 politician president;  
 city capital;  
 *int* population;  
 *void* readCountry(){  
 **cin**>>name;  
 president.readPolitician();  
 capital.readCity();  
 **cin**>>population;  
 }  
  
 *void* printCountry(){  
 **cout**<<name<<" ";  
 president.printPolitician();  
 capital.printCity();  
 **cout**<<population<<endl;  
 }  
  
 *int* countrycmp(country other){  
 *return* capital.population>other.capital.population;  
 }  
};  
  
*int* main(){  
 *int* **n**;  
 **cin**>>**n**;  
 country **c**[100];  
 *for*(*int* **i**=0 ; **i**<**n** ; **i**++){  
 **c**[**i**].readCountry();  
 }  
 country **max**;  
 **max**=**c**[0];  
 *for*(*int* **i**=1 ; **i**<**n** ; **i**++){  
 *if*(**c**[**i**].countrycmp(**max**)){  
 **max**=**c**[**i**];  
 }  
 }  
 **cout**<<"The president of the country whose capital city has the largest population is:"<<endl;  
 **cout**<<**max**.president.pName;  
 *return* 0;  
}