CS2310 final exam question

Computer Programming (City University of Hong Kong)

CITY UNIVERSITY OF HONG KONG

Course code & title	:	CS2310 Computer Programming
Session	:	Semester A 2021/22
Time allowed	:	Two hours

This paper has 7 pages (including this cover page).

1. This paper consists of 5 questions

2. Answer ALL questions in the space provided.
   (Students may use pencil if necessary).

3. Unless otherwise stated, libraries other than **<iostream>** and **<iomanip>** *(e.g. string class)* are NOT allowed

Students are allowed to use the following materials/aids:

Any hardcopy material

Materials/aids other than those stated above are not permitted. Students will be subject to disciplinary action if any unauthorized materials or aids are found on them.

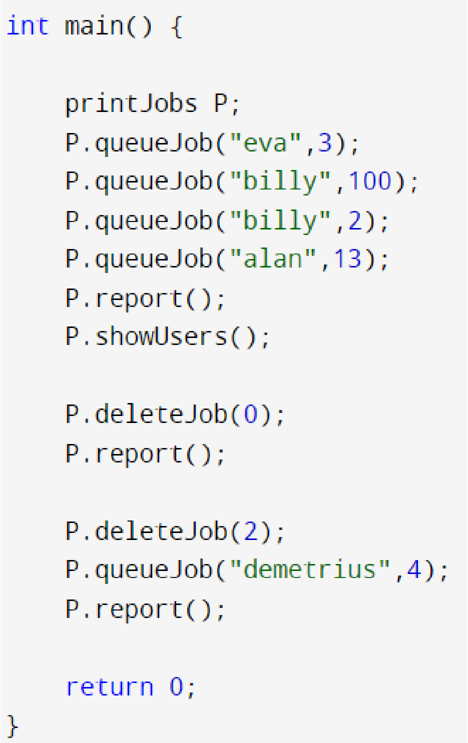
Student ID: ____________________

Q1(37)	Q2(13)	Q3(12)	Q4(18)	Q5(20)	Total(100)

1. In this question, you'll create a **printJobs** class to record the print jobs *(user name and number of pages)* issued to a shared printer. The class contains the following private members:

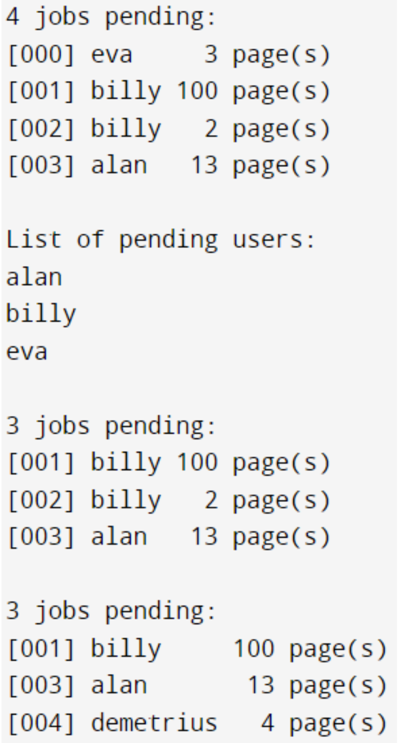
`int nJobs;`	Number of records (max 100) in **ID[]**, **user[]** and **nPage[]** mentioned below
`int ID[100];`	Automatic, unique IDs of the print jobs
`char user[100][16];`	Names of the users who have issued the print jobs (max 15 characters)
`int nPage[100];`	Number of pages in the print jobs

Using only the **<iostream>**, **<iomanip>** and **<cstring>** libraries, complete the public functions specified in a) to d). You may create local variables in the functions but you are not allowed to add in extra private/public data members. The sample **main()** function and the corresponding output is given to you below. You may refer to the figure for the detailed format and behaviour.



```
int main() {

    printJobs P;
    P.queueJob("eva",3);
    P.queueJob("billy",100);
    P.queueJob("billy",2);
    P.queueJob("alan",13);
    P.report();
    P.showUsers();

    P.deleteJob(0);
    P.report();

    P.deleteJob(2);
    P.queueJob("demetrius",4);
    P.report();

    return 0;
}
```

main() function



```
4 jobs pending:
[000] eva     3 page(s)
[001] billy 100 page(s)
[002] billy   2 page(s)
[003] alan   13 page(s)

List of pending users:
alan
billy
eva

3 jobs pending:
[001] billy 100 page(s)
[002] billy   2 page(s)
[003] alan   13 page(s)

3 jobs pending:
[001] billy     100 page(s)
[003] alan       13 page(s)
[004] demetrius   4 page(s)
```

Corresponding Program Output

a). Implement the **queueJob** function which stores the **user name** and **number of pages** in the internal arrays if the queue is not full. It also stores the automatic job **ID**, which is one plus the largest **ID** in the print queue *(start from 0 when the queue is empty)*. [5 marks]

b). Implement the **deleteJob** function which remove the print job record with the specified **ID**. The function returns a Boolean to indicate whether the removal is successful *(i.e. **ID** found)*. Upon successful removal, job records following the deleted record should shift up. [7 marks]

c). Implement the **report** function which display the number of jobs followed by the job details. Each job record begins with the 3-digit **ID**, followed by the **username**, which is *left-aligned* in the field width of the longest name. After that is the **pages** count, which is *right-aligned* according to the maximum value in the queue. [13 marks] d). Implement the **showUsers** function which display the name(s) of the user(s) in the print queue. The function should display the name(s) in *ascending order with no duplication*. This function should not change the **user[]** member in the **printJobs** and creation of new char/string array is not allowed. [12 marks]

2. In the space provided, write down the output of the following program. [8 marks]
With no more than 20 words, explain the purpose of function **F**. [5 marks]

```
#include <iostream>
using namespace std;

void F(int A[], int sA, int B[], int sB) {
  int i=0,j=0;
  while (i<sA && j<sB) {
    if (A[i]==B[j]) {
     i++, j++; continue;
    }
    if (A[i]<B[j]) cout << A[i++]<<" ";
    else           cout << B[j++]<<" ";
  }
  while (i<sA) cout << A[i++]<< " ";
  while (j<sB) cout << B[j++]<< " ";
}

int main() {
  int A[] = {1,2,3,4,5,6,7,8,11};
  int B[] = {1,3,5,8,9,10,13};
  F(A,9,B,7);
  return 0;
}
```

Program Output:

Purpose of function F:

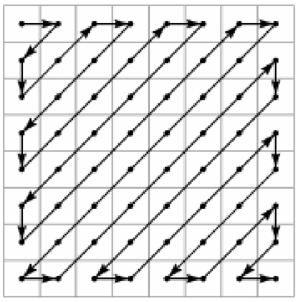
3. With recursion, derive whether the digits in integer **P** is a *sub-sequence* of the digits in another integer **N**. *i.e. derive whether **N** contains all the digits in **P** <u>in order (but not necessarily continuous)</u>* . You may assume that both numbers are positive. [12 marks]

   <u>In this question, students are not allowed to use loop, array or any library.</u>

Input	**Output**	**Explanation**
`12345   24`	Y	`12345 contains 2 & 4 and 2 comes before 4.`
`12345   42`	N	`Although 12345 contains 4 & 2, but 2 comes first.`
`12345   22`	N	`12345 contains only a single 2.`
`12345   12345`	Y	`12345 contains all digits in correct order.`
`1  1234567890`	N	`N contains only one digit.`

```
#include <iostream>
using namespace std;

bool check(int N, int P) {


















}
int main() {
    int N, P;
    cin >> N >> P;
    if (check(N,P))  cout << "Y\n";
    else             cout << "N\n";
    return 0;
}
```

4. In modern image / video compression schemes like JPEG2000 and H.264, it is quite common that the transformed image data is being scanned in a zig-zag order. The scan starts from the upper left corner, then it moves towards the lower-left direction. When the scan touches the boundary, it turns over and moves towards the opposite direction *(e.g. from lower-left to upper-right).*
   Students may refer to the diagram on the right.



   In this question, students will work on an enhanced version, which scans a *rectangle* in any arbitrary size *(0 < Width, Height < 21)*. The program should indicate the scan order using characters '**A**' to '**Z**' *(after 'Z', it will be 'A' again)*. [18 marks]

Example1 ***(user input underlined)***	**Example2** ***(user input underlined)***	**Example3** ***(user input underlined)***

`Input H and W: 5 12`
`ABFGOPYZIJST`
`CEHNQXAHKRUB`
`DIMRWBGLQVAC`
`JLSVCFMPWZDG`
`KTUDENOXYEFH`
	`Input H and W: 11 3`
	`ABF`
	`CEG`
	`DHL`
	`IKM`
	`JNR`
	`OQS`
	`PTX`
	`UWY`
	`VZD`
	`ACE`
	`BFG`
		`Input H and W: 4 4`
		`ABFG`
		`CEHM`
		`DILN`
		`JKOP`

5. Implement a function called **Cap()** which format the input cstring as specified [20 marks]:

- □ The function allocates memory for the output cstring and the start pointer is returned.
- □ The function is not allowed to change the content of the input cstring.
- □ The function treats all non-alphabet character as space and alphabet sequence as word.
- □ The output cstring contains no leading and trailing spaces and the word(s) are separated by exactly one space character.
- □ The words will be rendered in lowercase, except for the first and last character of each word, which is in uppercase.
- □ Students are not allowed to use any library in the solution

```
#include <iostream>
using namespace std;

char *Cap(char *S) {



}
int main() {
    cout << "[" << Cap("hELLo123") << "]\n";                  //[HellO]
    cout << "[" << Cap("?? thIS is~  a 'TEST'??!") << "]\n";  //[ThiS IS A TesT]
    return 0;
}
```

-- END OF PAPER --