THE UNIVERSITY OF HONG KONG

FACULTY OF ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE

ENGG1340 Computer Programming II
COMP2113 Programming Technologies

Date: May 18, 2020
Time: 2:30pm – 4:30pm

Instructions

This paper carries a total of 100 marks and consists of THREE sections: A, B and C.

Section A (30%) contains 15 multiple choice questions.

Section B (30%) contains 8 short questions.

Section C (40%) contains 2 programming questions.

Answer ALL questions.

Students are permitted to bring to the examination one sheet of A4-sized paper with printed or written notes on both sides.

Write your university number on the first page of your answer script.

Write your answers on paper. Scan your finished scripts as a single PDF file and submit to OLEX.

Use of calculator is NOT allowed.

Section A. Multiple Choice Questions. (30%)

There are altogether 15 questions in this part. Select a single answer for each question.

On your answer script, write down the letter of your answer for each question.

---

1. What command is used to count the total number of lines, words, and characters contained in a file?
   A. `wc`
   B. `countw`
   C. `wcount`
   D. None of the above

2. What command is used to sort the lines of data in a file in alphanumerical order?
   A. `sort -r`
   B. `sh`
   C. `sort`
   D. None of the above

3. What command is used to copy directory structures in and out?
   A. `copy`
   B. `cpio`
   C. `cp -p`
   D. None of the above

4. How to print the content of file "`1.txt`"?
   A. `cat 1.txt`
   B. `print 1.txt`
   C. `echo 1.txt`
   D. None of the above

5. If you don't know the options of command "`more`", what command will show you its options and descriptions?
   A. `help more`
   B. `man more`
   C. `google more`
   D. None of the above

6. The expression `if (num != 65)` cannot be replaced by:
   A. `if (num > 65 || num < 65)`
   B. `if (!(num == 65))`
   C. `if (num - 65)`
   D. `if (!(num - 65))`

7. Which one is the correct way to insert comments in C/C++ code?
   A. `// This is a comment`
   B. `/* This is a comment`
   C. `# This is a comment`
   D. `""" This is a comment """`

8. The function prototype

```
double mySqrt(int x);
```

   A. Declares a function called `mySqrt` which takes an integer as an argument and returns a `double`.
   B. Defines a function called `double` which calculates square roots.
   C. Defines a function called `mySqrt` which takes an argument of type `x` and returns a `double`.
   D. Declares a function called `mySqrt` which takes a `double` as an argument and returns an integer.

9. What does the function `mystery` do?

```
int mystery (int number)
{
   if (number <= 1) {
      return 1;
   }
   else {
      return number * mystery(number - 1);
   }
}
```

   A. Contains syntax error and does not compile
   B. Goes into infinite computations without return
   C. Always returns 1 regardless of the input
   D. Calculates the factorial of the input `number`

10. In C, which method can be used to find the length of a string?
   A. `len()`
   B. `strlen()`
   C. `getSize()`
   D. `length()`

11. All of the following can cause a fatal run-time error except:
   A. Dereferencing a pointer that has not been assigned to point to a specific address.
   B. Dereferencing a pointer that has not been initialized properly.
   C. Dereferencing a null pointer.
   D. Dereferencing a variable that is not a pointer.

12. Assuming that `t` is an array and `tPtr` is a pointer to that array, which expression refers to the address of element `3` of the array?
    A. `*(tPtr + 3)`
    B. `tPtr[3]`
    C. `&t[3]`
    D. `*(t + 3)`

13. Consider the statement:

    ```
    string* x, y;
    ```

    A. `x` is a pointer to a string, `y` is a string
    B. both `x` and `y` are pointer to string types
    C. `y` is a pointer to a string, `x` is a string
    D. none of above

14. In C/C++ program, the `#include` directives are expanded during?
    A. Run time
    B. Debug time
    C. Compile time
    D. Coding time

15. If `string s1` has the value “`computer`” and `string s2` has the value “`promise`”, which call to `insert` will produce the string `"compromise"`?
    A. `s1.insert(4, s2, 0, string::npos);`
    B. `s1.insert(string::npos, s2, 0, 4);`
    C. `s2.insert(0, s1, 0, 3);`
    D. `s2.insert(3, s1, 0, 3);`

– End of Section A –

Section B. Short Questions. (30%)

There are altogether 8 questions in this section. Answer ALL questions.

B1. (3%) Write down the output of the following code segment.

```
int x = 3;
int y = 2;
int z = 1;
if (z / 3 == 0 && x % 3 == 0 && y % x == 1) {
    x+=1;
}
y += x / 2;
cout << x * y * z;
```

B2. (3%) Write down the output of the following code segment.

```
void f(int a) {
    a = a*a;
    {
        int a = 4;
        cout << ++a << " ";
    }
    cout << a << " ";
}

int main() {
    int a = 1;
    {
        cout << a << " ";
        int a = 2;
        f(a);
        cout << a << " ";
    }
    f(a);
    return 0;
}
```

B3. (4%) Write down the output of the following code segment.

```
int f(char n[]) {
    int j = 0;
    for (int i = 0; n[i] != '\0'; ++i){
        int k = n[i] - '0';
        j = j * 10 + k + 1;
    }
    return j;
}

int main() {
    char n[] = "2019";
    cout << f(n);
    return 0;
}
```

B4. (4%) Write down the output of the following code segment.

```
void f(int p[], int size)
{
    int *a = &p[1];
    for (int i = size-2; i >= 0; --i)
        a[i] += a[i-1];
}

int main()
{
    int a[] = {6, 5, 4, 3, 2, 1};

    int * q = &a[2];
    f(q, 3);

    for (int i = 0; i < 6; ++i)
        cout << a[i] << ' ';

    return 0;
}
```

B5. (4%) Finish this task using a combined command (a.k.a. piping) using four individual commands. *Task*: Find all lines containing the keyword “`wonderful`” from file “`day.txt`” and sort the lines alphabetically. Pick the first fifteen of the sorted lines and count how many characters there are in the fifteen lines.

B6. (4%) Suppose you have an integer variable `x` that holds a four-digit positive integer. Write a **single** C++ statement to store the middle two digits of `x` in integer variable `y`.

B7. (4%) Consider the following code segment:

```
line 1:   int main()
line 2:   {
line 3:     int* p1, p2;
line 4:     int n = 42;
line 5:     p1 = p2 = &n;        //You cannot change this line
line 6:     printf("%d", *p1);   //You cannot change this line
line 7:   }
```

The expected output of the above code segment is 42, but the code cannot compile. Propose a fix which involves changes in line 3 and/or line 4 only.

B8. (4%) Consider the following code segment:

```
line 1:   int main()
line 2:   {
line 3:     const int SIZE = 42;
line 4:     double *p;
line 5:     if ((p = (double*)malloc(sizeof(double)*SIZE)) == NULL)
line 6:     { exit(EXIT_FAILURE); }
line 7:     for (int i = 0; i < SIZE; i++)
line 8:     { *(p + i) = i; }
line 9:     double *index = p + SIZE;
line 10:    double lastVal = *index;    //You cannot change this line
line 11:    printf("%lf\n", lastVal);  //You cannot change this line
line 12:    return 0;
line 13:  }
```

The expected output of the above code segment is 41, but the code does not run correctly. Propose a fix. Specify clearly the line(s) for the change.

– End of Section B –

Section C. Programming Questions. (40%)

There are altogether 2 questions in this part. Answer ALL questions.

On your answer script, start a new page for this question.

C1 (20%)

"Binary Search" searches a sorted array of elements stored in increasing values, by repeatedly dividing the search interval in half. The initial search interval covers the whole array. The search key is compared against the item at the middle of the interval (e.g., the middle of array [1,2,3] is 2, the middle of array [1,2,3,4,5,6] is 3). If the value of the search key is less than the item at the middle of the interval, we shrink the interval to its lower half. Otherwise, we shrink the interval to its upper half. So basically, half of the elements in the interval will be ignored after each comparison. We do this repeatedly until the search key *x* is found, or the interval is empty.

You need to provide two implementations of the Binary Search method, one being an iterative version using while loop , and the other using recursion. The main function and the function header of the Binary Search method have been written for you.

The function header of the Binary Search method:

```
int BinarySearch(int arr[], int l, int r, int x);
```

where `arr[]` is the input sorted array with elements stored in increasing values, `l` and `r` give the leftmost and rightmost array indexes (inclusively) of the interval to be searched, and `x` is the search key. If `x` can be found in `arr[]`, the function returns its index in the array; if `x` cannot be found in `arr[]`, it returns `-1`.

The main function:

```
int main(void)
{
    int arr[] = { 2, 3, 4, 10, 40 };
    int n = sizeof(arr) / sizeof(arr[0]);
    int x = 10;
    int index = BinarySearch(arr, 0, n - 1, x);
    if (index == -1)
        printf("Element is not present in array")
    else
        printf("Element is present at index %d", index);
    return 0;
}
```

(a) Write the **iterative version using while loop** for the BinarySearch function (10%).

```
int BinarySearch(int arr[], int l, int r, int x) {

    // Write the complete function (including the function header)
    // on your answer script






}
```

(b) Write the **recursive version** for the BinarySearch function (10%).

```
int BinarySearch(int arr[], int l, int r, int x) {

    // Write the complete function (including the function header)
    // on your answer script






}
```

C2 (20%)

Consider a reservation system for surgical mask purchases that is be able to

- add a new order to the order list (function `order`),
- output the order list from most recent to least recent order (function `output`),
- remove the oldest order from the list (function `deliver`).

The following C++ code employs a linked list for the reservation system.

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

struct Mask {
  string type;
  string customer;
  Mask *next;
};

void order(Mask *&head, string type, string customer) {
  cout << "Ordering " << type << " for " << customer << endl;
  Mask *oldHead = head;
  head = new Mask;
  head->type = type;
  head->customer = customer;
  head->next = oldHead;
}

void output(Mask *head) {
  cout << "Outputting order list " << endl;
  for (Mask *p = head ; p != NULL ; p = p->next)
    cout << "  " << p->type << " for " << p->customer << endl;
}

void deliver(Mask *&head) {
  cout << "Delivering " << head->type;
  cout << " for " << head->customer << endl;
  Mask *newHead = head->next;
  delete head;
  head = newHead;
}

int main() {
  Mask *head = NULL;
  order(head, "3M-N95" , "Alice");
  order(head, "OxyAir" , "Bruce");
  order(head, "3M-N95" , "Cindy");
  output(head);
  deliver(head);
  output(head);
}
```

This is the output of the program:

```
Ordering 3M-N95 for Alice
Ordering OxyAir for Bruce
Ordering 3M-N95 for Cindy
Outputting order list
  3M-N95 for Cindy
  OxyAir for Bruce
  3M-N95 for Alice
Delivering 3M-N95 for Cindy
Outputting order list
  OxyAir for Bruce
  3M-N95 for Alice
```

Unfortunately, the function `deliver` removes the newest order in the reservation list instead of the oldest one. A correct implementation of `deliver` would result in the following output.

```
Ordering 3M-N95 for Alice
Ordering OxyAir for Bruce
Ordering 3M-N95 for Cindy
Outputting order list
  3M-N95 for Cindy
  OxyAir for Bruce
  3M-N95 for Alice
Delivering 3M-N95 for Alice
Outputting order list
  3M-N95 for Cindy
  OxyAir for Bruce
```

(a) Rewrite the function deliver such that it will remove the oldest order in the list instead of the newest one. You may only change the body of the function deliver. (10%)

```
void deliver(Phone *&head) {

    // Write the complete function (including the function header)
    // on your answer script









}
```

(b) Due to problems with oversea shipping of the 3M-N95 model, it can no longer be delivered. Write the body of the following function, which will remove all orders in the list for which the type is “3M-N95”. (10%)

```
void remove(Phone *&head) {

    // Write the complete function (including the function header)
    // on your answer script









}
```

– End of Section C –

– END OF PAPER –