0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

(Spend 5 minutes to complete this part.)

A. Add semi-colons where appropriate:

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

void main()
{
    int x
    cout << "What is the sum of 200 and 9? Your answer: "
    cin >> x
    if (x==200+9)
    {
        cout << "Well done!"
        << endl
    } else
    {
        cout << "It is wrong."
        <<endl
    }
}
```

(Also learn how the else-part is designed.)

B. Here is a program segment similar to Lab01 Q04. Fill in the missing quotes and symbols.

```cpp
char choice ;
cout Please input your choice: ;
cin choice ;
if ( choice w ) ...
```

C. Will the code segments below display the same output as `cout << "ok";`? If no, what will be displayed? Guess!

```cpp
cout << "o" << "k";
cout << 'o';
cout << 'k';
char x;
char y;
x='o';
y='k';
cout << x;
cout << y;
```

```cpp
cout << "o" << "k";
cout << 'o';
cout << 'k';
char x;
char y;
x='o';
y='k';
cout << 'x';
cout << 'y';
```

```cpp
cout << "o" << "k";
cout << 'o';
cout << 'k';
char c1;
char c2;
c1='o';
c2='k';
c2=c1;
c1=c2;
cout << c1;
cout << c2;
```
D. After compilation, we seek to have the result (Fill in the blanks):

E. Which one below will have compilation problem? __________ (a) or (b)

F. Have you seen the following compilation error? Why does it happen?

G. In PASS, if the result is NOT 100% correct, we should click to check what's wrong in every test case. Find out where we should click.

For each test case, PASS has an expected output. PASS will execute your program and check whether the execution result matches the expected output.

H. What are whitespaces?

I. If you have any question in this course (eg. cannot install ..., don't know how to write the program, ...), you can email to ________________
Lecture Demo Exercises

Q1. In the Temperature Program, what will occur if we change `C_temp` and `F_temp` to `int` (instead of `double`)?

Q2. What will be the compilation error if our code contains the statements like:

```
int x;
3+4=x;
```

Answer: _______________________________________

Q3. What is the output of the following program?

```
.. void main()
  {
    int x;
    double y;
    x=3;
    y=x+2.5;
    x=y+2.5;
    cout << x << y << endl;
  }
```

Q4. An example of using `bool`.

(Program based on Lab01_Q04).

The program shows "Wrong input" if we type a wrong choice.

Like setting integer variables to 1,2,3..., We can set Boolean variables to `true` and `false`.

The logic:
- At the beginning, initialize a flag to `false`.
- If the input is valid, set the flag to `true`.
- In the end, if the flag is still `false`, we say "Wrong input".

Complete the code ____________________________________

```
.. void main()
  {
    char choice;
    bool bInputCorrect;
    cout << "Input your choice ('w' or 'h'): ";
    cin >> choice;
    bInputCorrect=__________;
    if (choice=='w')
    {
      cout << "Welcome World!";
      cout << endl;
      bInputCorrect=__________;
    }
    if (choice=='h')
    {
      cout << "Hello World!";
      cout << endl;
      bInputCorrect=__________;
    }
    if (bInputCorrect=__________)
    {
      cout << "Wrong input.";
      cout << endl;
    }
  }
```

This name is good for the flag because it clearly describes the usage (meaningful).
Q5. Extend the `swap` program to shift 3 values:

```cpp
.. void main()
{
    int value1, value2, value3;

    cout << "Input value1, value2, and value3: ";

    cout << "After shifting: " << value1 << " " << value2 << " " << value3 << endl;
}
```

Q6. (a) Complete the following code which shows the hour in 12-hours format (based on a 24-hours input).

```cpp
.. void main()
{
    int hr;

    cout << "Input the hour in 24-hours format: ";
    cin >> hr;

    cout << "The hour in 12-hours format is: ";

    cout << ":00";
    cout << endl;
}
```

(b) Redo the above using one expression only (for A++ students only):

```cpp
.. void main()
{
    int hr;

    cout << "Input the hour in 24-hours format: ";
    cin >> hr;

    cout << "The hour in 12-hours format is: " << ___________ << ":00";
    cout << endl;
}
```
Q7. (a) Type conversion between int and char

The char type is internally a 1-byte integer. Conversions are automatically done in some cases:

Examples:
- `cout << 'f'-'a';` shows 5.
- `cout << 'f'-5;` shows 97.
- `char x=65;` sets x to 'A'.
- `int x='A';` sets x to 65.
- `'A'<='B'` gives true.

Question:

What will be shown when we run the code below?

```cpp
void main()
{
    char c1,c2,c3,c4,c5;
    cout << "Input a 5-letters word (in lowercase): ";
    cin >> c1 >> c2 >> c3 >> c4 >> c5;
    /* Hint: calculate c1-'a' to give the "distance" between c1 and 'a'. Then add this distance as an offset to 'A'=> uppercase. */
    c1 = _______________________________; // Hint: calculate c1-'a' to give the "distance" between c1 and 'a'. Then add this distance as an offset to 'A'=> uppercase.
    c2 = _______________________________; // Hint: calculate c2-'a' to give the "distance" between c2 and 'a'. Then add this distance as an offset to 'A'=> uppercase.
    c3 = _______________________________; // Hint: calculate c3-'a' to give the "distance" between c3 and 'a'. Then add this distance as an offset to 'A'=> uppercase.
    c4 = _______________________________; // Hint: calculate c4-'a' to give the "distance" between c4 and 'a'. Then add this distance as an offset to 'A'=> uppercase.
    c5 = _______________________________; // Hint: calculate c5-'a' to give the "distance" between c5 and 'a'. Then add this distance as an offset to 'A'=> uppercase.
    cout << "The uppercase version is: ";
    cout << c1 << c2 << c3 << c4 << c5 << endl;
}
```

(b) Changing from lowercase to uppercase by shifting ASCII code!
Q8. In the Compound Interest Program, if the starting amount is changed to 900.00, the output is not well aligned. Try to modify the code to improve it (also change the precision to 1):

<table>
<thead>
<tr>
<th>Year</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>936.0</td>
</tr>
<tr>
<td>2</td>
<td>973.44</td>
</tr>
<tr>
<td>3</td>
<td>1012.38</td>
</tr>
<tr>
<td>4</td>
<td>1052.87</td>
</tr>
<tr>
<td>5</td>
<td>1094.99</td>
</tr>
</tbody>
</table>

Press any key to continue . . .

Q9. Refer to the Hourly Session Program in the lecture notes, how to improve it so that the case 23:00-00:00 can be shown correctly? [This question will be included in Lab02.]
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

Spend 5 minutes for the questions in this part. See how much you can finish. 😊

A. (i) If you find difficulties in take-home exercises, you should:
   [Put ☑ or ☐ for each choice.]
   - read the question again
   - check hints from Guided-exercises or redo the Guided-ex again
   - read Helena’s email and check course web
   - contact helpdesk (available from course web)
   - revision and reading (lecture notes and text-book)
   - try changing the code here and there and wait for luck
   - next time I should start doing exercises earlier (not last minute)

(ii) True/False [Put ☑ or ☐ below.]
   - "My code is better than you because I use less variables." ☐
   - "My code is better than you because my logic is simple and clear. It is easy to read and maintain." ☑
   - "My code is better than you because it is short and tricky." ☐

B. True/False [Put ☑ or ☐ below.]
   - Every "cin <<" pauses for my input until I press <enter>. ☑

C. The code below makes use of setw and setfill. Your task: Complete ① and ②:

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

void main()
{
    cout << "12345678901234567890" << endl;
    cout << "--------------------" << endl;
    cout << setw(4) << 89 << setw(10) << "Hello" << 3 << endl;
    cout << setfill('0') << setw(4) << 67;
    cout << setw(5) << 45 << setfill(' ') << setw(6) << 23 << endl;
}
```

By default:
- numbers are right-aligned and
- text are also right-aligned

What will be shown in the output?

```
Output:
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
= = = = = = = = = = = = = = = = = = = = = = =
Press any key to continue .. .
```
D. Given the code below, **trace** by hands and fill in the table.

```cpp
void main()
{
    int a, b, c, d;
    a=-10;
    a=a+1;
    b=a;
    b++;
    c=-b;
    d=c%3;
    d+1;
    cout << a << " " << b << " " << c << " " << d << endl;
}
```

**Values of variables:**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>after a=-10:</td>
<td>-10</td>
<td>[value undefined]</td>
<td>[value undefined]</td>
</tr>
<tr>
<td>after a=a+1:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>after b=a:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>after b++:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>after c=b:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>after d=c%3:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>after d+1:</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The output will be: 

D. Given the code below, suppose the user types "3<space>3<enter>", what will be the output?

```cpp
void main()
{
    int i, x;
    char c;
    cin >> i >> c;
    x=c;
    cout << i << " " << c << " " << x << endl;
}
```

```cpp
void main()
{
    int a,b,c,d;
    a=-10;
    a=a+1;
    b=a;
    b++;
    c=-b;
    d=c%3;
    d+1;
    cout << a << " " << b << " " << c << " " << d << endl;
}
```

The output will be: 

F. (i.) Fix the error of the following using **full braces** and improve it with **correct indentations**.

```cpp
if (choice=='w')
    cout << "Welcome!";
    cout << endl;
else
    if (choice=='h')
        cout << "Hello!";
        cout << endl;
    else
        cout << "Invalid choice!";
        cout << endl;
```

(ii.) "If the executed sequence involves only 1 statement, then {} are optional in most cases."

Your task: **Rewrite** the code with **no braces**.

For your reference - ASCII table

<table>
<thead>
<tr>
<th>Characters</th>
<th>'0'</th>
<th>'1'</th>
<th>...</th>
<th>'9'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>48</td>
<td>49</td>
<td>...</td>
<td>57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Characters</th>
<th>'A'</th>
<th>'B'</th>
<th>...</th>
<th>'Z'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>65</td>
<td>66</td>
<td>...</td>
<td>90</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Characters</th>
<th>'a'</th>
<th>'b'</th>
<th>...</th>
<th>'z'</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>97</td>
<td>98</td>
<td>...</td>
<td>125</td>
</tr>
</tbody>
</table>
Lecture Demo Exercises

Q1. Consider the code:

(i) What is the structure of the above program code as seen by the compiler?  

- An if-statement nested inside an if-else statement
- An if-else statement nested inside an if-statement
- An if-statement followed by an if-else statement
- None of the above

(ii) The code can't behave correctly in some cases. Describe one such case: when temperature is

(iii) How to correct it?

(iv) "In order to make our code easy to follow, we should attempt to handle the easiest case first."

How would you further improve the code above?

Q2. Consider the code:

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

void main()
{
    int s;
    cout << "Input your score: ";
    cin >> s;
    if (s >= 85)
        cout << "A";
    else if (s >= 70) //line 12
        cout << "B"; //line 13
    else if (s >= 60) //line 14
        cout << "C"; //line 15
    if (s >= 60)
        cout << " (PASSED)" << endl;
    else
        cout << "F (Failed)" << endl;
}
```

(i) Complete the output in the cases below:

Write down the expected output here.

<table>
<thead>
<tr>
<th>Input your score: 70</th>
<th>Input your score: 59</th>
</tr>
</thead>
<tbody>
<tr>
<td>Press any key to continue . . .</td>
<td>Press any key to continue . . .</td>
</tr>
</tbody>
</table>

Write down the expected output here.
(ii) If we exchange lines 12-13 and lines 14-15 like:

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

void main()
{
    int s;
    cout << "Input your score: ";
    cin >> s;
    if (s >= 85)
        cout << "A";  //line 14
    else if (s >= 60)  //line 15
        cout << "C";
    else if (s >= 70)  //line 12
        cout << "B";  //line 13
    if (s >= 60)
        cout << " (PASSED)" << endl;
    else
        cout << "F (Failed)" << endl;
}
```

Any problem? Show it below:

```
Input your score: 70

Press any key to continue . . .
```
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

Spend 5 minutes for the questions in this part. See how much you can finish. 😊

Using while-loop

A. Use a while-loop to show all multiples of 19 which are smaller than 100.

```java
void main()
{
    int n = 19;
    while (n < 100)
    {
        System.out.println(n);
        n = n + 19;
    }
}
```

B. Use a while-loop to find the smallest multiple which is larger than 1000.
The program should give the following result:

```java
void main()
{
    int n = 1000;
    while (n < 10000)
    {
        n = n + 19;
    }
}
```

C. Homework / Self-drilling

Use a while-loop to find the largest multiple which is under 10000.
The program should give the following result:

```java
void main()
{
    int n = 10000;
    while (n > 19)
    {
        System.out.println(n);
        n = n - 19;
    }
}
```

There are many approaches to do this exercise. What's your friend's approach?

** There exist mathematical ways to do B and C (without using a loop).
However, please do practice well using loops ➔ An important aim of this chapter.
D. Function

The following program sets Mary's score to 52 and Paul's score to 49.
Bonus of 10 scores is given if score < 50.

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

// Return the resultant score after adjustment with bonus
int adjust_with_bonus(int score)
{
    if (score < 50)
        return score+10;
    else
        return score;
}

void main()
{
    int mary_score,paul_score,new_mary_score,new_paul_score;
    mary_score=52;
    paul_score=49;
    new_mary_score=adjust_with_bonus(mary_score);
    new_paul_score=adjust_with_bonus(paul_score);
    if (new_paul_score>new_mary_score)
        cout << "Not fair\n";
    else
        cout << "Okay\n";
}
```

(i) Study the program. What will be the output? Your answer [ A or B ] : _____________

(A) [Image of output showing "Not fair"]
(B) [Image of output showing "Okay"]

(ii) At the end of execution, the variable values are:

    mary_score:   _________  paul_score:   _________
    new_mary_score:  _________  new_paul_score:  _________

(iii) The function can be changed as shown on the right.
The program also works.

Recall: A function parameter (here score) acts like variables. In addition, it is initialized with the value passed from the caller function (here main).

```cpp
int adjust_with_bonus(int score)
{
    if (score < 50)
        score+=10;
    return score;
}
```

(a) What is the relationship between score and mary_score, and between score and peter_score?
Answer:

(b) Though allowed, but "changing the value of a parameter in the function for convenience of coding" is not so good.
Justify the above saying:
E. Rewrite (a) in (b) only

\[
\begin{array}{|c|c|c|}
\hline
(a) & (b) & (c) \\
\hline
\text{bool noHomework=true;} & \text{bool existHomework=} & \text{bool existHomework=} \\
\text{if (num\_MA\_course\_Homework>0)} & ={} & \text{if (num\_MA\_course\_Homework=} \\
\text{noHomework=false;} & \text{existHomework=} & \text{existHomework=} \\
\text{if (num\_CS2331\_Homework>0)} & ={} & \text{if (num\_CS2331\_Homework=} \\
\text{noHomework=false;} & \text{existHomework=} & \text{existHomework=} \\
\text{if (num\_EN\_Homework>0)} & ={} & \text{if (num\_EN\_Homework=} \\
\text{noHomework=false;} & \text{existHomework=} & \text{existHomework=} \\
\text{if (noHomework!=true) } & \text{cout << "Disappointed!!"} & \text{cout << "Disappointed!!"} \\
\hline
\end{array}
\]

Lecture Demo Exercises

Q1. Which one(s) of the \texttt{hex2dec} functions below have "Not all control paths return a value" problem?

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

void main()
{
    // .. read a hex digit into a char variable and call one of the above functions
    // .. assume that the hex digit is a valid one: '0'-'9' or 'A'-'F'
}
```

```cpp
int hex2dec_V1(char c)
{
    if (c>='0'&&c<='9')
        return c-'0';
    else if (c>='A'&&c<='F')
        return 10+(c-'A');
}
```

```cpp
int hex2dec_V2(char c)
{
    if (c>='0'&&c<='9')
        return c-'0';
    else
        return 10+(c-'A');
}
```

```cpp
int hex2dec_V3(char c)
{
    if (c>='0'&&c<='9')
        return c-'0';
    return 10+(c-'A');
}
```

Q2. Write down the expected output for the code below.

```cpp
int a=4;
int b=3;

cout << a  <<endl;
cout << b  <<endl;
cout << (a+b) <<endl;
cout << (a>b) <<endl; //given: shows 1
```
Q3. Different ways to give a boolean expression in the if-statement.

(a) Suppose \( n\text{Questions} \) stores the number of exercise to be done. Choose the correct (meaningful) ones below:

- if \((n\text{Questions}>0)\)
  
- if \((n\text{Questions}>=1)\)
  
- if \((n\text{Questions}-1>=0)\)
  
- if \(((n\text{Questions}>0) == true)\)
  
- if \((n\text{Questions}) \land \text{type conversion}!!\)
  
- if \(((n\text{Questions}) \land \text{type conversion}!!)\)

(b) Suppose we set one more variable: \( \text{bool existQuestion} = (n\text{Questions} > 0); \)

- if \((\text{existQuestion}==true)\)
  
- if \((\text{existQuestion})\)
  
- if \((\text{existQuestion}!=false)\)
  
- if \((\text{existQuestion}!=true)\)
  
- if \((!\text{existQuestion})\)

(c) Suppose we further added: \( \text{bool noQuestion} = (n\text{Questions} == 0); \)

- if \((\text{noQuestion}==false)\)
  
- if \((!\text{noQuestion})\)
  
- if \((\text{noQuestion}!=true)\)

Type conversion (Recall)

- From int to bool
  - Non-zero => true
  - Zero => false

- From bool to int
  - True => 1
  - False => 0

Q4. Write down the expected output for the code below.

(a) ```
#include <iostream>
using namespace std;

void main()
{
    int x=0;
    int i;
    for (i=0; i<4; i++)
    {
        cout << i << endl;
    }
}
```  

(b) ```
#include <iostream>
#include <iomanip>
using namespace std;

void main()
{
    int x=0;
    int i,j;
    for (i=0; i<4; i++)
    {
        for (j=0; j<3; j++)
        {
            cout << setw(3) << x
            x=x+2;
            cout << setw(3) << i
            cout << setw(3) << j;
            cout << endl;
        }
    }
}
```
Casual Discussion, Warm-up Questions

Suppose now we need to write a longer program (say totally 70 lines or more), choose the correct one(s) below:

1. To complete the program in shortest time, I can write all steps in a long main function first. Before I submit it to the boss, I will split it part-by-part into several functions.
   - Any disadvantage(s)? _______________________________________________

2. To complete a program in shortest time, the fastest way is to apply the top-down approach according to the idea covered in week 4 lessons (lecture+lab).
   - The main function should be short and contain least details. Through designing the main function, I also plan for a list of required functions and I will create the functions later.
   - Any disadvantage(s)? _______________________________________________

Lecture Demo Exercises

Question 1. [Cont’d – Week 4 lecture exercise]
Which one(s) of the hex2dec functions below have the "Not all control paths return a value" problem?

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

int hex2dec_V4(char c)
{
    switch (c)
    {
        case 'A': case 'B': case 'C': case 'D': case 'E': case 'F':
            return 10+(c-'A'); // "break" is not needed because "return" means to stop the function.
        case '0': case '1': case '2': case '3': case '4':
            case '5': case '6': case '7': case '8': case '9':
                return c-'0';
    }
}

int hex2dec_V5(char c)
{
    switch (c)
    {
        case 'A': case 'B': case 'C': case 'D': case 'E': case 'F':
            return 10+(c-'A'); // "break" is not needed because "return" means to stop the function.
        default: // ie. cases '0' to '9'
            return c-'0';
    }
}

void main()
{
    // .. read a hex digit into a char variable and call one of the above functions
    // .. assume that the hex digit is a valid one: '0'-'9' or 'A'-'F'
}
```
Question 2. Recall the example used in Week 3 lecture (Page 9): (Lec03_P9_function_hexDigit.cpp)

Now rewrite it with one function only (see comments below):

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

/* Attempt to convert a given hexadecimal digit character (h - call-by-value) to
decimal value (result - call-by-ref)
If success, return true, otherwise return false. */

bool hex2dec(________ h, ________result)
{
    if (h>='0' && h<='9')
    {
    }
    else if (h>='A'&&h<='F')
    {
    }
    else
    {
    }

    void main()
    {
        char c;

        cout << "Input a hexadecimal digit: ";
        cin >> c;

        if
            cout << "Its value in decimal is: " << << endl;
        else
            cout << "It is not a hexadecimal digit" << endl;
    }

    --- end ---
```
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions
Spend 5 minutes for the questions in this part. See how much you can finish. 😊

A. Read the conversations below. Who are correct?

Peter says, "I don’t want to use the debugger. I want to check the bug by studying the code. If I can’t find the bug, I will throw away my current code and start over again."

Paul says, "Now I’m writing short programs. This is a good time to practice using the debugger. So that later I will be able to use debugger effectively in large programs."

Mary says, "I cannot use the debugger during exam. So now, whenever I have a chance, I should use the debugger to help me solve more program bugs and finish more programs, and learn from them."

B. Matching

1. Global variables
2. static – static variables
3. return – statement
4. & – Call-by-reference
5. Comments before every function

- We need to learn what they are. But not recommended to use
  "if not necessary, don’t use"
- Important
We will use them a lot in CS2331

C. Are the followings the same? Which one may cause different result?

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>for (int i=0; i&lt;100; ++i)</td>
</tr>
<tr>
<td></td>
<td>{ cout &lt;&lt; i; }</td>
</tr>
<tr>
<td>2</td>
<td>int i;</td>
</tr>
<tr>
<td></td>
<td>for (i=0; i&lt;100; ++i)</td>
</tr>
<tr>
<td></td>
<td>cout &lt;&lt; i;</td>
</tr>
<tr>
<td>3</td>
<td>for (int i=0; i&lt;100;)</td>
</tr>
<tr>
<td></td>
<td>{ cout &lt;&lt; i;</td>
</tr>
<tr>
<td></td>
<td>i++;</td>
</tr>
<tr>
<td>4</td>
<td>int i=0;</td>
</tr>
<tr>
<td></td>
<td>for (; i&lt;100; ++i)</td>
</tr>
<tr>
<td></td>
<td>{ cout &lt;&lt; i; }</td>
</tr>
<tr>
<td>5</td>
<td>int i=0;</td>
</tr>
<tr>
<td></td>
<td>for (i; i&lt;100; i++)</td>
</tr>
<tr>
<td></td>
<td>{ cout &lt;&lt; i; }</td>
</tr>
</tbody>
</table>

D. What will be shown if the followings are executed? Why?

Ref: Lec04.pdf
- Un-intentional Empty Statements]

- 0 1 2 3 4 5 6 7 8 9
- 0 1 2 3 4 5 6 7 8 9 10
- 10
- Nothing is shown (the loop can’t stop)
E. Call-by-reference

Consider the following main function:

```cpp
void main()
{
    int i1=4;
    int i2=18;
    int i3=9;
    int i4=10;
    swap(i1,i2);
    swap(i3,i4);
    cout << i1 << " " << i2 << " " << i3 << " " << i4 << endl;
    reverse_order_of_4(i1,i2,i3,i4);
    cout << i1 << " " << i2 << " " << i3 << " " << i4 << endl;
}
```

Write the 2 functions: `swap` and `reverse_order_of_4` according to how the main function calls them:

```cpp
//swap the contents of 2 input arguments.
//try make use of the swap function

//reverse the contents of 4 input arguments.
//try make use of the swap function
```

F. Read the conversations below. Who are correct?

Peter says, "Call-by-reference is powerful than call-by-value. If I'm not sure, I may just put & before every parameter, eg. bool hex2dec(char &sh, int &result)"

Paul says, "We should design data types, identifiers' names, and parameter passing mode appropriately. They should help us understand what the code does and minimize confusions and errors."

Mary says, "If I want to use a function but it has a call-by-reference parameter, I'll assume that the function will take my variable and update it. If actually I don't expect such an update, then I'll doubt about whether the function is suitable. So please don't abuse call-by-reference."
Lecture Demo Exercises

Question 1.
Complete the `swap_front_and_end` function on the right such that it can be called to Exchange the first and last elements in an array.

```cpp
// swap the contents of 2 input arguments.
void swap(int &x1, int &x2)
{
    int temp;
    temp=x1;
    x1=x2;
    x2=temp;
}

// read integers from console and store in an array
void swap_front_and_end(int arr[], int size)
{
    // Note: use the swap function
    swap(_______________________________________);
}

void main()
{
    int marks[5]={51,52,53,54,55};
    swap_front_and_end(marks,5);
}
```

Question 2.
Design a function which can be called to reverse all contents in an array.

```cpp
// swap the contents of 2 input arguments.
void swap(int &x1, int &x2)
{
    int temp;
    temp=x1;
    x1=x2;
    x2=temp;
}

// read integers from console and store in an array
void reverse_contents(int arr[], int size)
{
    // for (____________________________________)
    swap(________________________________________);
}

void main()
{
    int marks[5]={51,52,53,54,55};
    reverse_contents(marks,5);
}
```
Question 3.
Design a function which can be called to find the index of a given student ID in a student list (integer array).
For example, if the array contains the data below, and the caller wants to find 50100041, then it is found with index equal to 3.

```
50100018 50100022 50100030 50100041 50100059 50100068 50100073
```

The function should scan the array elements **sequentially** and return Boolean (true/false) based on whether the ID is found. The found index (if any), should be obtained by the caller through a call-by-reference parameter.

Write the function below:

```c
//Sequential search : Find a given value in an array of integers.

__________ sequential_search(_______________________________) {

}
```
Question 4. Hardcoding and Readability of code

Introduction

Explanation:
- A named constant is not a variable: it can’t be changed.
  Eg. if later we accidentally type PI=x*4; compiler will give error (Then it is safe! Because harmful code cannot compile + run)
- We say: "Instead of hardcoding (3.14) everywhere, we define and use a named constant (PI)."

Reason to apply a named constant (instead of hardcoding everywhere):

(a) Suppose we want to change 3.14 to 3.1416:

Instead of changing all 3.14, now only need to change once (eg. Line 4 const double PI=3.1416;)

(b) We want to make the program easier to read (eg. using WIDTH instead of 800 everywhere)

(c) Minimize mistakes, eg. 3.14 .. 3.14 .. 3.14 .. 3.74 hard to debug!!

Test your understanding

Which styles (i–v) will you choose?

(i) Packing Oranges

```
const double PI=3.14;
const int WIDTH=800;

double calCirclePerimeter(double r)
{ return r*2*PI; }

double calCircleArea(double r)
{ return r*r*PI; }

// apply PI also in other places ..
void main()
{ // ..
}
```

Defines a named constant PI. Then we can type PI to use its value everywhere.

(ii) Packing Oranges

```
const int BOX_SIZE=12;

n= tot/ BOX_SIZE;
r= tot% BOX_SIZE;
```

Program A : Not preferred – hardcoding 3.14 everywhere

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

double calCirclePerimeter(double r)
{ return r*2*3.14; }

double calCircleArea(double r)
{ return r*r*3.14; }

// apply 3.14 also in other places ..
void main()
{ // ..
}
```

Program B : Improved by applying a named constant

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

const double PI=3.14;
const int WIDTH=800;

double calCirclePerimeter(double r)
{ return r*2*PI; }

double calCircleArea(double r)
{ return r*r*PI; }

// apply PI also in other places ..
void main()
{ // ..
}
```

(iii) hex to dec

```
value1 = 10 + (c1 - 65);
```

What is 65? Is it easy to understand?

(iv) hex to dec

```
value1 = 10 + (c1 - 'A');
```

Using ‘A’ is easier to understand than using 65. Right?

Even if we change to a new coding scheme later (not ASCII), we don’t need to change this program code!!

(v) hex to dec

```
const char charA='A';

value1 = 10 + (c1 - charA);
```

Comparatively, (iv) is easier to read than (v). Because ‘A’ can be directly understood.

Defining charA is unnecessary (not supported by any reason among (a), (b), (c) – see ** above).
Question 5. In page 9 of the lecture notes, the program in example 3 produces sample runs (i) and (ii). Now we want to rewrite it for sample run (iii):

Sample run (iii):

<table>
<thead>
<tr>
<th>Input the marks, ended with -1: 51 52 -1</th>
</tr>
</thead>
<tbody>
<tr>
<td>After adding bonus: 61 62</td>
</tr>
<tr>
<td>Press any key to continue ...</td>
</tr>
</tbody>
</table>

Corresponding to Sample run (iii), the main function is rewritten as:

```cpp
void main()
{
    int marks[MARK_ARR_SIZE], i, n;

    cout << "Input the marks, ended with -1: ";
    input_into_arr(marks, n);
    for (i=0; i<n; i++)
        marks[i]+=10;
    cout << "After adding bonus: ";
    for (i=0; i<n; i++)
        cout << marks[i] << " ";
    cout << endl;
}
```

**Your task:** Rewrite the `input_into_arr` function according to how it is called by the main function above. Try **2 different ways** as indicated below:

### Version 1: Use a do-while-loop
(Each time read one value, if not -1, store to the array, continue as long as the value is not -1).

```cpp
/* Read integers from console and store in an array (1st parameter : array) 
Also update the effective size of the array (2nd parameter: call-by-reference int) */

void input_into_arr(______________)
{
    int value;

    do
    {
        cin >> value;

        if (value!=-1)
        {
        }
    }
    while (value!=-1);
}
```

### Version 2: Use a while-loop
(Input first value. Then loop as long as the value is not -1. Read next value at the end of the loop to prepare for next iteration.)

```cpp
/* Read integers from console and store in an array (1st parameter : array) 
Also update the effective size of the array (2nd parameter: call-by-reference int) */

void input_intoArr (______________)
{
    int value;

    cin >> value;

    while (value!=-1)
    {
        cin >> value;
    }
}
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

A. Complete the following function which can be called to delete an element in an array of characters.
   - The index of the unwanted element, `idDelete`, is given as a function parameter.
   - If `idDelete` is invalid (ie. not within 0..n-1), simply return false
   - If `idDelete` is valid, remove the element by
     - firstly replace the contents in `arr[idDelete..n-2]` by `arr[idDelete+1..n-1]`
     - then update `n` by `n-1`.

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

//From the array arr which contains n integers, delete the element at position idDelete

void deleteElement(char arr[], int n, int idDelete)
{

}

void main()
{
    char a[7]={'P','r','o','g','r','a','m'};
    int n=7;

    if (deleteElement(a, n, 4))
        cout << "Successful.\n\n";  
    else
        cout << "Wrong index.\n\n";

    cout << "Final content: ";
    for (int i=0;i<n;i++)
        cout << a[i] << " ";
    cout << "\n\n";
}
```

Draw initial array contents:

Draw final array contents:

```
Successful.
Final content: P r o g r a m
Press any key to continue . . .
```
B. Suppose an array contains some elements in ascending order, now one more item (an intruder) is added at the end but it is not the largest value.

\[
\text{Eg. } \{10, 15, 18, 23\} \quad \text{add 17 (an intruder)}
\]

Your task: Finish the following function to fix the order of all elements (Based on the given comments).

Note: Refer to how it is called by the `main` function.

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

// swap 2 values
void swap(int &v1, int &v2)
{
    int original_v1;
    original_v1 = v1;
    v1 = v2;
    v2 = original_v1;
}

/* Suppose a[0..n-2] contains sorted elements, a[n-1] is a new "intruder"
   Attempt to insert a[n-1] among a[0..n-2] so as to fix the order of the overall sequence */

   fix_new_intruder(______________)
{
}

void main()
{
    int n;
    int a[10]={10, 15, 18, 23}; // a[0] to a[3] are already sorted
    a[4]=17; // add an arbitrary value at the end (a[4]);
    n=5; // now it contains 5 elements: 10, 15, 18, 23, 17

    fix_new_intruder(a,n);

    cout << "Result: ";
}
```

Draw initial array contents:

```
{ 10, 15, 18, 23, 17 }
```

Draw final array contents:

```
{ 10, 15, 18, 23, 17 }
```

Result: 10 15 17 18 23
Press any key to continue . . .
Lecture Demo Exercises

Question 1. Complete the following program which copies contents from a 2D array to a 1-D array.
(Note how the main function calls other functions.)

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

const int Arr2D_Width=5;
const int Arr2D_Height=3;

// put random values in a 2D array (A given function)
void setRandomContents(int arr[][Arr2D_Width])
{
    for (int i=0;i<Arr2D_Height;i++)
        for (int j=0;j<Arr2D_Width;j++)
            arr[i][j]=rand()%20; // a random number from 0 to 19 (We will learn random numbers later.)
}

// display contents in a 2D array
void displayArr2D(_____________________________)
{
}

// Copy contents from a 2D array to a 2D array
void copyFrom2DTo1D(________________________________________)
{
}

void main()
{
    int arr2D[Arr2D_Height][Arr2D_Width]; // 2D
    int arr1D[Arr2D_Height*Arr2D_Width]; // 1D

    setRandomContents(arr2D); // Given function : put random values in arr2D
    cout << "Original Contents:\n";
    displayArr2D(arr2D); // <= You are to write this function
    cout << endl;

    copyFrom2DTo1D(arr2D,arr1D); // <= You are to write this function
    cout << "Resultant contents in the 1D array:\n";
    for (int i=0;i<Arr2D_Height*Arr2D_Width;i++)
        cout << arr1D[i] << " ";
    cout << endl;
}
```

Original Contents:
1 7 4 1 0 9 0 8 1 8 2 4 5 5 1 7 1
Resultant contents in the 1D array:
1 7 1 4 0 9 4 1 8 2 4 5 5 1 7 1

Press any key to continue . . .
Question 2. Complete the following program which reads the contents of n values from a file and shows the sum.

```cpp
#include <_______________>
#include <_______________>
using namespace std;

void main()
{
    ifstream inFile;
    int n;
    int sum=0;
    // The following "array of char" (string) is to accept users' typing of the file pathname
    // user may input pathnames like: "Infile1.txt", "c:\myFolder\output.txt" etc..
    // Note that the size of this array must be 1 larger than the actual length which the user types.
    char inFile_pathname[200];
    cout << "Type the file pathname for input: ";
    cin >> inFile_pathname;
    // Open the file

    // Read n from the file

    for (int i=0; i<n; i++) // Read values and add to the sum
    {

    }

    cout << "The sum of the values is: " << sum << endl;

    // close the file
}
```

n is 5
5 values to be summed

We will learn about strings 2 weeks later.
Question 3. The following program reads pairs of values from a file and output the sums to another file.

Your task: Fill in all missing code according to the comments.

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

void main()
{
    ifstream inFile;
    ofstream outFile;
    char inFile_pathname[200];
    char outFile_pathname[200];
    cout << "Type the file pathname for input: ";
    cin >> inFile_pathname;
    cout << "Type the file pathname for output: ";
    cin >> outFile_pathname;

    // Open the input file. If "fail", show message and exit(1);
    if (                            )
    {
        cout << "Can’t open input file.\n"
    }

    // Open the output file. If "fail", show message and exit(2);
    if (                            )
    {
        cout << "Can’t open output file.\n"
    }

    // Read pairs from the input file until eof.
    // If not failed, write the sum to the output file.
    // If "fail" to write, show message and exit(3);

    // Close the files
}
```

Note that there may be blank line(s) at the end of the file.

[To open and read such a file based on a typed file pathname, refer to page 10 of the lecture notes.]
Question 4. Refer to question 3, rewrite the loop for file reading without using `eof`. (ie. directly check the value of `inFile >> x1 >> x2`)

Refer to page 10 of the lecture notes

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

void main()
{
    ifstream inFile;
    ofstream outFile;
    char inFile_pathname[200];
    char outFile_pathname[200];
    cout << "Type the file pathname for input: ";
    cin >> inFile_pathname;
    cout << "Type the file pathname for output: ";
    cin >> outFile_pathname;

    // Open the input file. If "fail", show message and exit(1);
    if (                            )
    {
        cout << "Can't open input file.\n";
    }

    // Open the output file. If "fail", show message and exit(2);
    if (                            )
    {
        cout << "Can't open output file.\n";
    }

    // Read pairs from the input file until no more successful input can be made.
    // write the sum to the output file.
    // If "fail" to write, show message and exit(3);

    // Close the files
}
```
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

A. Which one below can pass result from one function to his caller?
   - return value       - call-by-reference       - global-variable       - write to a file (and let the caller reads the file)

   Suppose the result is only 1 simple value, which one in the above is preferred?

B. Compare the 2 functions below. Which one is "more useful"?

   Design a function which can be used to find the HCF of 2 numbers:

   ```
   /* Return the maximum value found in the array. */
   int getMaxCount(const int arr[], int size)
   {
      int mx=0;
      for (int i=0; i<size; i++)
         if (arr[i]>mx)
            mx=arr[i];
      return mx;
   }
   
   /* Show the maximum value found in the array. */
   void showMaxCount(const int arr[], int size)
   {
      int mx=0;
      for (int i=0; i<size; i++)
         if (arr[i]>mx)
            mx=arr[i];
      cout << "Maximum value is: " << mx << endl;
   }
   ```

   A simple approach:
   * Note a special case: if x1 is 0, return x2 (and vice versa)
   * Otherwise, it is normal and easy:
     Suppose x1 is 18 and x2 is 27, which one is smaller? Answer: 18
     So set i to 18.
     Then, while x1 is not divisible by i or x2 is not divisible by i, try next smaller i (ie. i--)
     (That is, the loop will stop when both x1 and x2 are divisible by i, so return i.)

   Advanced solution (for interested students only): Euclidean algorithm

C. Short questions

   (i) Suppose `int data[3][4][5]` is to represent some data in 3D (several pages, each page has some rows/columns). Which one of the below is the best match?
(ii) We can use binary search on unsorted lists. (True/False)

(iii) The following is correct (True/False):

```
[Inside our program code]
ifstream inFile;
inFile.open("c:\m.txt");
```

(iv) Mary said, "The following is wrong. When we type the file pathname, we should type `c:\q5_1.txt`." [When we type in the console]

Is Mary correct?

C. (i) What is the output given by the following program?

```java
void testing(int a[], int b[]) {
    for (int i=0; i<3; i++)
        b[i]=a[i];
    if (b[0]==b[1]&&b[1]==b[2])
        cout << "b[0], b[1], b[2] are the same" << endl;
    else
        cout << "b[0], b[1], b[2] have some difference" << endl;
}
void main() {
    int a[3]={10,10,10};
    int b[3];
    testing(a,b);
}
```

(ii) What will happen if we change Line A from `b[i]=a[i]` to `b[i]==a[i]`?

Note: - Usually our default compilation is in debug-mode.
- Release-mode compiled code may behave differently (This usually happens if we have programming mistake that causes "by-luck" behavior).

- In PASS, the compilation is release-mode.
- Well, indeed even with the same executable file, two different computers may still have chance of giving different results.
Lecture Demo Exercises

Question 1. Array of chars ("nearby" gifts).
A given data file contains the count of gifts in each zone of a rectangular field:

Note:  
- There are at most 10 rows and 10 columns.
- At most 9 gifts are stored in each zone.

Finish the following program reads the file. Then inputs a zone and counts the nearby gifts in the zone itself as well as its 8 neighboring zones.

(Use the debugger when needed: breakpoint + F5)

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

// Read the data of gifts into an array of char
void readGifts()
{
    ifstream inFile;
    inFile.open("c:\gifts.txt");

    // Read the data of gifts into an array of char
    void readGifts()
    {
        ifstream inFile;
        inFile.open("c:\gifts.txt");

        // Given a location, count the neighbouring gifts in 8 neighbouring zones
        int countNearbyGifts()
        {
        
    }

    void main()
    {
            int input_row, input_col;
            readGifts();
            cout << "Input a location \[row:0-""<<height-1"", col:0-""<<width-1"errals\": ";
            cin >> input_row >> input_col;
            cout << "It has "
                 << countNearbyGifts()
                 " gift(s) nearby.\n" ;
```
Question 2. More on the Date class

Based on Lec08_DateV2:

(i) Add 2 constructors:
   - Default constructor (set date to Jan-1, 2008)
   - Constructor with 3 parameters for year, month, and day

(ii) Suppose the program (main function) is to be rewritten and run as follows:

Enter year, month and day as numbers: 2008 10 21
You typed: year = 2008, month = 10, day = 21
The next day is: year = 2008, month = 10, day = 22
Press any key to continue . . .

Enter year, month and day as numbers: 2008 12 31
You typed: year = 2008, month = 12, day = 31
It is the end of a month.
The next day is: year = 2009, month = 1, day = 1
Press any key to continue . . .

Your task: Finish the 2 required functions:

/* Check whether the calling date object contains a
date that's the ending day of any month */
bool Date::isEndOfAMonth()
{
   //Hint: use the valid member function
}

/* Return the "tomorrow" of the calling date object */
Date Date::next()
{
   //Hint: use the isEndOfAMonth member function
}

/*Date.h*/

class Date
{
   public:
      void input();
      void output();
      void set(int new_y, int new_m, int new_d);
      int get_year();
      int get_month();
      int get_day();

   private:
      // check if y,m,d valid
      bool valid(int y, int m, int d);
      // check if y is a leap year
      bool isLeapYear(int y);
   
   int year, month, day;
};

/*srcMain.cpp*/

#include <iostream>
#include "Date.h"
using namespace std;

void main()
{
   Date d, dNext;
   d.input();
   cout << "You typed: ";
   d.output();
   if (d.isEndOfAMonth())
      cout << "It is the end of a month." << endl;
   else
      cout << "It is not the end of a month." << endl;

   dNext=d.next();
   cout << "The next day is: ";
   dNext.output();
}

/* Check whether the calling date object contains a
date that's the ending day of any month */
bool Date::isEndOfAMonth()
{
   //Hint: use the valid member function
}

/* Return the "tomorrow" of the calling date object */
Date Date::next()
{
   //Hint: use the isEndOfAMonth member function
}
0. Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

A. Review – Refer to the Coins class on the right:

- There is/are ____ constructor(s).
  The default constructor is: ________________.
- Including the constructors, there are ____ member functions.
- Among the member functions, the private member function(s) is/are: ________________.
- There are ______ data members (member variables).
- In the main function, circle the calling object where whose member functions are being called.
- In the input function, highlight the data members of the calling object.

Lecture Demo Exercises

Question 1. (Refer to Page 4 in Lecture notes.)

1. If we change the main() function as shown on the right, what will happen during the execution?

2. Now, further change
   
   ```cpp
   int pick_one(Big b)
   ```
   to
   
   ```cpp
   int pick_one(Big &b)
   ```
   What does this mean? What will happen during the execution?

3. What else can be made to improve the code?
Question 2. Random numbers

Introduction: We can use `rand()` to generate a random number (0..32767). Here it is a super die with values 0..32767!!

Think of a die with values 1-6.

Program A

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

void main()
{
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
}
```

Press any key to continue . . .

Your task: Rewrite the code to simulate a die with values 1-6. Roll it 10 times. ( Hint: use `%6` )

Program B

```
#include <iostream>
#include <time.h>
using namespace std;

void main()
{
    srand( (unsigned)time( NULL ) ); //set the seed
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
}
```

Press any key to continue . . .

To make it "really" random – that means to get different sequences when you run it at different times.

- When the program starts running, use the time value of that moment (an encoded integer) to set "an internal seed".

2 lines needed: `srand( (unsigned)time( NULL ) );` //set the seed

Your task: Do it.

Program C

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

void main()
{
    unsigned int seed=1;
    int rand()
    {
        unsigned int result = seed * 214013 + 2531011;
        seed=result;
        return (result/65536) % 32768;
    }

    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
    cout << rand() << endl;
}
```

Press any key to continue . . .

Tell you a secret: The code of `rand()` used by Microsoft in Visual Studio:

```
unsigned int seed=1;
int rand()
{
    unsigned int result = seed * 214013 + 2531011;
    seed=result;
    return (result/65536) % 32768;
}
```

- Unsigned int is also a fundamental type.
  - int: -2,147,483,648 to 2,147,483,647
  - unsigned int: 0 to 4,294,967,295

- result could be a very large value.
  - we copy result as the next seed for calculation.
  - we return a transformed value of result:

  ```
  return (result/65536) % 32768;
  
  ○ Create a smaller value from result.
  ○ Limit it to 0..32767
  ```

Make it "really" random:

- Looking back – Why program B always gives the same sequence: "6655651153", "6655651153", "66555..
Answer: Because the program always start with `seed=1` (see the secret above).
Casual Discussion, Warm-up Questions, and Lecture Demo Exercises

Casual Discussion, Warm-up Questions

A. [2008-09 Test 2] Write your answers for (a) and (b) in the empty spaces of the source code below.

(a) Overloaded the * operator as a friend function of the Fraction class so that we can multiply a fraction with an integer.

For example, it can be used by the following code segment.

(b) Add a member function half() to the Fraction class so that we can call it to compute the half of a fraction.

For example, it can be used by the following code segment.
(c) State and briefly describe the mistake(s) of the following function header for the extraction operator:

```cpp
friend Fraction operator>>(istream &ins, Fraction fract)
{
    ...
}
```

(B) Write a program which inputs a number of fractions into an array and find the largest one.
(5 marks)

(It should make use of the Fraction class built in Lab 9.)

The expected run-down is:

```cpp
#include "fraction.h"
#include <iostream>
#include <fstream>
using namespace std;
Fraction maxFraction(__________________________)
{
}
void main()
{
    int n;
    cout << "Input n (n<=10): ";
    cin >> n;
    cout << "Input " << n << " Fractions (must be positive):
";
    for (int i = 0; i < n; i++)
    {
        // Input fraction
    }
    cout << "Largest fraction is: ";
}
```

C:\WINDOWS\system32\cmd.exe
Input n (n<=10): 4
Input 4 Fractions (must be positive):
1/3
12/5
6/15
5/3
Largest fraction is: 12/5
Press any key to continue . . .
Lecture Demo Exercises

Question 1. Explicit type conversion for fundamental types

(just like using a conversion constructor / Lecture09.pdf page 7)

Syntax: \texttt{simple-type-name \ ( expression )}

Useful: We can convert fundamental values to the wanted types like:

\begin{lstlisting}[language=C++]
int totMarks=533;
int totCourses=7;
cout << "Average is : 
<< double(totMarks)/totCourses
<< endl;
\end{lstlisting}

Your task:
The program on the right attempts to read a letter and then show the next char (not the code) in the ASCII table.

However, it gives wrong output. Fix it.

\textbf{Expected output} \hspace{1cm} \textbf{Actual output}

\begin{tabular}{|c|c|}
\hline
Input a letter: d & Input a letter: d \\
The next letter in ASCII is: e & The next letter in ASCII is: 101 \\
Press any key to continue . . . & Press any key to continue . . . \\
\hline
\end{tabular}

Question 2. Significance of the null character \texttt{\textbackslash 0}’

Complete the code below:

- Output a cstring char by char
  \begin{lstlisting}[language=C++]
  void my_cout(const char s[], int n) /* Output the cstring s char by char */
  {
    int i=0;
    while (_______________)
    {
      cout << s[i];
      i++;
    }
  }
\end{lstlisting}

The above explains how \texttt{\textbackslash 0} in cstrings are actually applied in most operations.

- Display / Remove the last character in a cstring (ie. The character before \texttt{\textbackslash 0})

\begin{tabular}{|c|c|}
\hline
(i) void display_last(const char s[]) & (ii) void remove_last(const char s[])
{
}
\hline
\end{tabular}

Note: In real practice, we often pass cstrings as \texttt{char *s} or \texttt{const char *s} instead of \texttt{char s[]} or \texttt{const char s[]}.
We will know more in our next lesson.
0. Warm-up Discussion and Questions

Casual Discussion, Warm-up Questions

A. (i) We can declare arrays as [ * automatic / dynamic ] variable like

```
int c[20]; c[0]=4; c[1]=9; c[2]=8;
```

We can declare arrays as [ * automatic / dynamic ] variable like

```
int *c; c=new int[20]; c[0]=4; c[1]=9; c[2]=8;
```

[* Delete the above where appropriate.]

(ii) The program on the right has compilation error.

(a) Explain the problem.

(b) Rewrite it using dynamic memory allocation.

(c) Rewrite int *p=new int[n]; as 2 statements:

```
void f(int n)
{
    int arr[n];
    //.. do something
}
void main()
{
    int n;
    cin >> n;
    f(n);
}
```

(c) Rewrite int *p=new int[n]; as 2 statements:

Fill in the blanks (using hints on the right):
In the code above, p is a pointer. But you may find it easier to learn if you regard p as a __________ whose type is memory __________.

This operation allocates a portion of computer __________ to the program and returns the __________ of this memory so that __________ is set to the memory address. p can then be used as __________ as well.

By dereferencing p, __________ refers to the first element. More usefully, you can type __________ to refer to elements in the array so as to change or read the elements.

Later when the memory is not no more needed, we should __________ this memory by passing the address to the __________ operator.

(iii) Complete the function on the right that creates a dynamic integer array of n integers and puts 1,2,...n inside it. Also design how it passes back the array to the caller.

```
// creates a dynamic integer array of n integers and puts 1,2,...n inside it
________ createIntArr( int n )
{
}
```

Given words and terms:
custom, variable, address, cpu, memory, variable, address, *p, p, automatic variable, array name, &p, *p, p[...], allocate, dispose, new, delete
B. The function on the right is to copy non vowels from an input cstring s to the result cstring.

Rewrite it using pointer arithmetic to traverse and update the result cstring:

Add a pointer \texttt{char* pR} to traverse it.
(The integer variable \texttt{idR} should be removed.)

```c
// copy non vowels from an input string \texttt{s} to the result string
void copyNonVowels(const char s[], char result[])
{
    int idS=0, idR=0;
    while (s[idS]!='\0')
    {
        if (tolower(s[idS])!='a'&&
            tolower(s[idS])!='e'&&
            tolower(s[idS])!='i'&&
            tolower(s[idS])!='o'&&
            tolower(s[idS])!='u')
        {
            result[idR]=s[idS];
            idR++;
        }
        idS++;
    }
    result[idR]='\0';
}
```

C. What are the differences of these terms:

- 0
- `'0'`
- '\0'
- "0"

\textbf{Lecture Demo Exercises}

1. Explain the code:

```
string s1[10];
char s2[10];
```

2. Any compilation error?

```
string a0,a1;
a0=a1;
char b0[10], b1[10];
b0=b1;
char c0[10]="", c1[10]="";
cout << (c0==c1);
```

3. When one expression contains 2 or more operators, then

\textbf{The order of Evaluation depends on precedence level and associativity.}

```
Example:
int x = 12345 / (4 + 5 * 7 - 2);
- The precedence level of / and * are higher than the precedence level of + and -
- To override the above ordering, we add () for grouping.

↑ the precedence level of () is high
```

Your task: For each highlighted expression below, mark the steps with \(\textcircled{1}, \textcircled{2}\):

(i) \texttt{cout << 1234 / 100 / 10;}

(ii) \texttt{cout << 1234 * 60 \% 24; (note: the precedence levels of * and \% are the same)}

(iii) \texttt{int a,b; a = b = 10;}

Your task: Delete the wrong items below (*):

In * (i) / (ii) / (iii), we say that the associativity is left-to-right.
In * (i) / (ii) / (iii), we say that the associativity is right-to-left.

\textbf{When operators have the same precedence level, then the associativity rules of the operators decide the order of evaluation.}

\textbf{For other operators, see “Order of Evaluation - Precedence and Associativity” at the course web (under “Useful Tips + Notes”).}
0. Warm-up Discussion and Questions

A. Refer to the `showDigits` function below:

```cpp
// Show digits of n, each followed by a space
void showDigits(int n)
{
  if (n<10)
    cout << n << " ";
  else
  {
    int leading=n/10;
    int right_most=n%10;
    showDigits(leading);
    cout << right_most << " ";
  }
}
```

Suppose we set a breakpoint at line 14 and run it with the debugger, and input 5678.

(a) Which of the below will be shown in the call stack window when the execution comes to Line 14 for the first time?

(i) `showDigts(int n=56)` Line 14
(ii) `showDigits(int n=567)` Line 13
(iii) `showDigits(int n=5678)` Line 13
(iv) `main()` Line 23
(v) `main()` Line 23

(b) Following question (a), what will be the output at that moment?

```
Input n: 5678
Press any key to continue . . .
```

(c) During the execution of a program, all local variables (including call-by-value parameters) are kept in the data stack while the functions run.

Refer to the instance of execution as shown on the right. Which of the below reflects the data stack's usage?

(i) 3rd `showDigit's n, leading, and right_most`
    2nd `showDigit's n, leading, and right_most`
    1st `showDigit's n, leading, and right_most`

   main's n

   5678, 56, 7
   5678, 567, 8
   56, 5, 6

The function calls in main and the several robots have separate "office spaces"

(ii) 1st `showDigit()`
    2nd `showDigit()`
    3rd `showDigit()`

   main

   5678

All function calls to `showDigit` share one office space.
B. Suppose one of the following (based on Lab12) is given in the exam paper, what's your approach to write the code?

```
// (a) determine whether an integer x contains even digit(s) (0,2,4,6,8)
   containEven(x)
{

}

// (b) show digits of integer x reversely (each followed by a space)
   showDigitsReverse(x)
{

}

// (c) determine whether a given digit exists in an integer x.
   containDigit(digit)
{

}

// (d) count the number of digits in an integer x.
   countDigits(x)
{

}

// (e) return left-most digit in integer x
   leftMostDigit(x)
{

}
```