THE UNIVERSITY OF HONG KONG

DEPARTMENT OF COMPUTER SCIENCE

ENGG1340 / COMP2113

Computer Programming II / Programming Technologies

2021-22 Semester 2

About the Course

Dr. RB Luo

Teaching Team

	Team A	Team B
Primarily responsible of	Module 1, 2, 4, 9, 10 Five problem solving sessions Project Mid-term quiz and final exam	Module 3, 5, 6, 7, 8 Three assignments
Instructors	Dr. RB Luo (rbluo@cs.hku.hk)	Dr. Chenxiong Qian (cqian@cs.hku.hk)
Teaching Assistants	Mr. Tim C.K. Lai (chunkiu@hku.hk)	Mr. Kevin Y. K. Lam (yklam2@cs.hku.hk)
	Mr. Marco K. H. Wong (marcow12@hku.hk)	Ms. Shumin Li (smli@cs.hku.hk)
	Ms. Qiuhui Li (qhli@cs.hku.hk)	Mr. Yao Lai (lishumin@hku.hk)
	Mr. Yekai Zhou (ykzhou@connect.hku.hk)	Ms. Runjian Chen (rjchen@hku.hk)

* Consultation hours details are available in Moodle

I have already taken COMP1117 / ENGG1330 before

and here's ANOTHER programming course?

What is this course about?

- You'll learn about the various software development technologies and tools that a **competent computer science professional** should know and be good at

- To prepare you with **solid programming skill and background** to cope with the upcoming challenges in the fundamental and advanced courses in the Computer Science curriculum

- To get you to start developing professional practice in coding

Go check out "The Eight Levels of Programmers"

Major Topics

- Linux programming environment
- Version control
- C/C++ programming language

git

Course Format

- This is a self-learning course
  - The ability of self-learning a new programming language is itself a learning objective of this course
  - This is not supposed to be an easy course
- Module-based with **checkpoint submissions** as progress check
- Five Problem-solving sessions (tentatively in the 6^{th}, 9^{th}, 10^{th}, 11^{th}, and 13^{th} week)
- One **on-site** mid-term quiz
- One group project

A tentative course calendar

- Changes and new arrangements will be announced in the "News Announcement" forum
- Do not switch off the email notification of the "News Announcement" forum
- All announcements made in "News Announcement" forum are considered delivered

ENGG1340 / COMP2113 (2021-2022 2nd semester)

Wk	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Notes
1 Jan-16 Jan-22	16	17 M0 release	18	19	20	21 First Lecture M1 release	22	Add/drop: January 17, 2022 (Mon) (10:00 am) – January 31, 2022 (Mon) (1:00 pm) #1: Linux Environment
2 Jan-23 Jan-29	23	24	25	26	27	28 M2 release	29	#2: Shell Script + Version Control
3 Jan-30 Feb-5	30	31 CNY holidays	1	2	3	4	5	
4 Feb-6 Feb-12	6	7 CNY holidays	8	9	10	11 M3 release	12 M0, M1 due	#3: C/C++ Basics
5 Feb-13 Feb-19	13	14	15	16	17	18 M4 release	19 M2 due	#4: Makefile, Programming Style, Basic Debugging, Python/C++ Comparison
6 Feb-20 Feb-26	20	21 M3 PS sessions	22	23	24 A1 release	25 M5 release	26 M3 due	#5: Functions Assignment 1 (up to functions)
7 Feb-27 Mar-5	27	28	1	2	3	4 M6 release Release Proj. Req.	5 M4 due	#6: Arrays &Strings Release project requirements
8 Mar-6 Mar-12	6	7 Reading week	8	9	10	11	12	Reading week: March 7, 2022 (Mon) – March 12, 2022 (Sat)
9 Mar-13 Mar-19	13	14 M5 PS sessions	15	16 Holiday	17 A2 release	18 M7 release A1 due	19 M5 due	#7: Structs, File I_O & Recursion Assignment 2 (up to arrays, recursion)
10 Mar-20 Mar-26	20	21 M6 PS Sessions	22	23	24	25 Quiz 1	26 Project Proposal due	Onsite quiz 1 Project proposal due
11 Mar-27 Apr-2	27	28 M7 PS sessions	29	30	31	1 M8 release	2 M6 due	#8: Pointers & Memory Management
12 Apr-3 Apr-9	3	4	5 Holiday	6	7 A3 release	8 M9 release A2 due	9 M7 due	Assignment 3 (up to pointers)
13 Apr-10 Apr-16	10	11 M8 PS Sessions	12	13	14	15 Holiday	16 Holiday	
14 Apr-17 Apr-23	17	18 Holiday	19	20	21	22 M10 release	23 M8 due	#9: C Specifics and GDB debugger
15 Apr-24 Apr-30	24	25	26	27	28	29 A3 due	30 M9 due	#10: Standard Template Library
16 May-1 May-7	1	2 Holiday	3	4	5	6	7 Project due M10 due	Revision period: May 2-7
17 May-8 May-14	8	9	10	11	12	13	14	Assessment weeks: May 9-28

Five Problem-solving sessions

- Tentatively in the 6th, 9th, 10th, 11th, and 13th week
- Time change is unlikely, but if necessary, will be announced in Moodle earliest possible
- **Fully online and mandatory**. Attendance will be taken
- Each student will be assigned to a group of 25-35, the grouping will be announced by the 4th week
- Time slots
  - Monday 12:30-14:20 ENGG1340B1, ENGG1340B2
  - Tuesday 10:30-12:20 COMP2113B, COMP2113C
  - Tuesday 13:30-15:20 ENGG1340C1, ENGG1340C2
  - *If you have another class time-clashed with your assigned problem-solving session time slot, make a request to TA Kevin with screenshot proofs. We might assign you to another time slot*
  - *Requests after Jan 31st will not be considered*
  - *Requests without a proof will not be considered*
  - *Emails badly written will not be considered*
- Students TAs will work through selected coding problems during the session

Ten modules

- Ten modules
  - #1: Linux Environment
  - #2: Shell Script + Version Control
  - #3: C/C++ Basics
  - #4: Makefile, Programming Style, Basic Debugging, Python/C++ Comparison
  - #5: Functions
  - #6: Arrays, Strings
  - #7: File I/O & Structures & Recursion
  - #8: Pointers & Memory Management
  - #9: C Specifics and GDB debugger
  - #10: Standard Template Library

Ten modules

- Learning materials with clear objectives will be provided
  - Learning materials & notes
  - Practice exercises for self-review
  - Readings & references
  - Checkpoint tasks (require submission and w/deadline)
- You need to complete the module checkpoint tasks and submit to Moodle before a given deadline for each module

Three assignments

- Programming assignments that you should **complete individually** and submit by the deadline
- Three weeks to complete
- **Read assignment instructions carefully and start early.**
- Deadlines **are strictly enforced**
  - **Late Policy: if submit within 3 days after the deadline, 50% deduction. After that, no mark**
  - **Exceptions will be considered case by case for only medical reasons or formal leaving**

Assignment Grading

- No credit for a program that cannot compile or run in a standardized environment (CS' server "academy11")
  - It is possible that your code runs fine in your own computer or servers but doesn't run in academy11. Always compile and test run your code in academy11 before submission
  - Learn to write code that is not specific to a particular programming environment is a learning objective of the course
- Your code will be auto-graded for technical correctness. Sample test cases are provided but additional test cases are used for grading
- Your project code will be graded for appropriate coding styles too
- Rebuttal is allowed for possible technical errors or misunderstandings
  - Please send your rebuttal request to your responsible student TAs (you will soon know the assignments), they will work out a decision with the lecturers
  - Rebuttal should be done in a week after the release of scores

Mid-term quiz

- On-site individual written test
  - Mar 25th, 2022 at CPG-LG.1
- Closed book
- Question types:
  - Multiple choice questions
  - Short questions
  - Programming questions
- There will be NO make-up or supplementary quiz. If no-show, the worst score among the three assignments will be used for projecting the quiz score
- Subject to the fluid situation of the pandemic, the quiz might be canceled. If so, other parts in the continuous assessment will be scaled up proportionally

Project

- You will work in a group of two, but grading will be done on an individual basis according to:
  1. Individual contribution tracked by GitHub
  2. Peer review
- You will work on the project for 7-8 weeks, with intermediate milestones to complete
- You will need to come up with specific ideas about a given topic
- At the end, you would have finished a complete project which incorporates the skills you learned from this course
- More details to be announced according to the calendar

Assessment

- Continuous Assessment (70%)
  - Problem-solving Sessions Attendance [5%]
  - Ten Checkpoint Task Submissions [10%]
  - Three Programming Assignments [30%]
  - One Programming Project [15%]
  - One Mid-term Quiz [10%] (contingent)
- Final Written Examination (30%)

On Plagiarism

- Discussions of materials and problems are encouraged. But make sure that your submission is your own work
- Plagiarism detection software will be used in the grading of every assignment
- Acts that would be considered as committing plagiarisms:
  - Copying (in part or in full) from peers, senior students, internet sources (no matter they are open source or private)
  - Sharing your work to peers
- You should learn how to best protect your own work from potential leak

On Plagiarism

- The teaching team is responsible for maintaining assessment fairness among all students, and so do you
- This is about academic and professional ethics, so we take it very seriously

I used my classmate's computer to code and left my work there. He accidentally uploaded my code as his submission

You should protect your work from potential leak and make sure you don't submit anything which is not your own work

There are classical problems which are good coding exercises. Since you are submitting for assessment purposes, you are required to submit your own work

The coding problem is typical, and I found answer from the Internet, so I just copy and submit it

Tips on how to avoid plagiarism:

- Don't look for too many references from the Internet
- Just get some idea and don't read into the code. Think on your own
- Don't share your code to others (including posting to anywhere where others can access)
- Code on your own

Plagiarism Policy

- As defined in the University's Regulations Governing Conduct at Examinations, plagiarism is the **unacknowledged use, as one's own, of work of another person, whether or not such work has been published**. Or put it simply, plagiarism is **copying the work of another person without proper acknowledgement**. In case of queries on plagiarism, students are strongly advised to refer to "What is Plagiarism?" (https://tl.hku.hk/plagiarism/)
- **First Attempt**: Students who admit committing plagiarism for the first time shall be warned in writing and receive a **zero mark for the component concerned**. For those who do not confess, the case would be referred to the Programme Director for consideration
- **Subsequent Attempt**: If students commit plagiarism more than once during the course of studies, the case shall be referred to the Programme Director for consideration. The Programme Director will investigate the case and consider referring it to the University Disciplinary Committee, which may impose any of the following penalties: **a published reprimand, suspension of study for a period of time, fine, or expulsion from the University**

Plagiarism example 1

- Change code sequence won't work

```
#include <stdio.h>
#include <iostream>

//function to check if the number is palindrome
[redacted]

bool isPalindrome(int x)
{
    int numb = x;
    int rever = 0;
    while (x != 0)
    {
        //to find the reverse of the number
        rever = rever * 10 + x % 10;
        x /= 10;
    }
    if (rever == numb) //to check if the reverse is equal to the original number
    {
        return true;
    }
    else
    {
        return false;
    }
}
//function to check if the number is a product of two 3 digit numbers
bool isProduct(int x)
{
    int prod;
    for (int i = 999; i >= 100;i--) //highest 3 digit number=999 and lowest=100
    {
        for (int j = i; j >= 100; j--)
        {
            prod = i * j;

            if (prod == x)
            {
                return true;
            }
        }
    }
    return false;
}
//main function to implement all functions
int main()
{
    using namespace std;
    int P, Q, i;
    char opt;
    bool pali, prod;
    cin >> P >> Q >> opt;

    switch (opt)
    {
    case 'p':               // if only palindromes need to be printed

        for (i = P;i <= Q;i++)
        {
            pali = isPalindrome(i);
            if (pali == true)
            {
                cout << i << endl;
            }
        }
        break;
    case 't':               // if only product of 2 3-digit numbers is to be printed

        for (i = P;i <= Q;i++)
        {
            prod = isProduct(i);
            if (prod == true)
            {
                cout << i << endl;
            }
        }
        break;
    case'b':                // if number is both palindrome and product of 2 3-digit numbers

        for (i = P;i <= Q;i++)
        {
            prod = isProduct(i);
            pali = isPalindrome(i);
            if (prod == true && pali == true)
            {
                cout << i << endl;
            }
        }
        break;
    }
    return 0;
}
```

```
#include <stdio.h>
#include <iostream>

//function to check if the number is palindrome
[redacted]

bool isPalindrome(int x)
{
    int num = x;
    int rev = 0;
    while (x != 0)
    {
        //finding the reverse of the number
        rev = rev * 10 + x % 10;
        x /= 10;
    }
    if (rev == num) //checking if the reverse is equal to the original number
    {
        return true;
    }
    else
    {
        return false;
    }
}
//function to check if the number is a product of two 3 digit numbers
bool isProduct(int x)
{
    int product;
    for (int i = 999; i >= 100;i--) //highest 3 digit number=999 and lowest=100
    {
        for (int j = i; j >= 100; j--)
        {
            product = i * j;

            if (product == x)
            {
                return true;
            }
        }
    }
    return false;
}
//main function to implement all functions
int main()
{
    using namespace std;
    int M, N, i;
    char opt;
    bool pal, prod;
    cin >> M >> N >> opt;

    switch (opt)
    {
    case 'p':               // if only palindromes need to be printed

        for (i = M;i <= N;i++)
        {
            pal = isPalindrome(i);
            if (pal == true)
            {
                cout << i << endl;
            }
        }
        break;
    case 't':               // if only product of 2 3-digit numbers is to be printed

        for (i = M;i <= N;i++)
        {
            prod = isProduct(i);
            if (prod == true)
            {
                cout << i << endl;
            }
        }
        break;
    case'b':                // if number is both palindrome and product of 2 3-digit number

        for (i = M;i <= N;i++)
        {
            prod = isProduct(i);
            pal = isPalindrome(i);
            if (prod == true && pal == true)
            {
                cout << i << endl;
            }
        }
        break;
    }
    return 0;
}
```

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

bool isPalindrome(int x);
int count_digit(int number);
bool isProduct(int x);

int main(){
    int m,n;
    char opt;
[redacted]

    cin >> m >> n >> opt;
    switch (opt)
    {
    case 'p':
        for (int i = m; i < (n+1); i++)
        {
            if(isPalindrome(i))
            {
                cout << i << endl;
            }
        }
        break;
    case 't':
        for (int i = m; i < (n+1); i++)
        {
            if(isProduct(i))
            {
                cout << i << endl;
            }
        }
        break;
    case 'b':
        for (int i = m; i < (n+1); i++)
        {
            if(isPalindrome(i) && isProduct(i))
            {
                cout << i << endl;
            }
        }
        break;
    }
    return 0;
}

[redacted]

bool isPalindrome(int x){
    int xO = x;
    int xR = 0;
    int loop = count_digit(x);
    for (int i = 0; i < loop; i++){
        xR = xR*10 + x%10;
        x = x/10;
    }
    if(xO == xR){
        return true;
        //cout << "Match!" << endl;
    } else{
        return false;
        //cout << "Not Match!" << endl;
    }
}

int count_digit(int number) {
   int count = 0;
   while(number != 0) {
      number = number / 10;
      count++;
   }
   return count;
}

bool isProduct(int x){
    for(int i=100; i<1000; i++){
        if(count_digit(x/i)==3 && x%i == 0){
            return true;
        }
    }
    return false;
}
```

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

bool isPalindrome(int x);
int count_digit(int number);
bool isProduct(int x);

[redacted]

bool isPalindrome(int x){
    int xO = x;
    int xR = 0;
    int loop = count_digit(x);
    for (int i = 0; i < loop; i++){
        xR = xR*10 + x%10;
        x = x/10;
    }
    if(xO == xR){
        return true;
        //cout << "Match!" << endl;
    } else{
        return false;
        //cout << "Not Match!" << endl;
    }
}

int count_digit(int number) {
   int count = 0;
   while(number != 0) {
      number = number / 10;
      count++;
   }
   return count;
}

bool isProduct(int x){
    for(int i=100; i<1000; i++){
        if(count_digit(x/i)==3 && x%i == 0){
            return true;
        }
    }
    return false;
}


int main(){
    int m,n;
    char opt;
[redacted]

    cin >> m >> n >> opt;
    switch (opt)
    {
    case 'p':
        for (int i = m; i < (n+1); i++)
        {
            if(isPalindrome(i))
            {
                cout << i << endl;
            }
        }
        break;
    case 't':
        for (int i = m; i < (n+1); i++)
        {
            if(isProduct(i))
            {
                cout << i << endl;
            }
        }
        break;
    case 'b':
        for (int i = m; i < (n+1); i++)
        {
            if(isPalindrome(i) && isProduct(i))
            {
                cout << i << endl;
            }
        }
        break;
    }
    return 0;
}
```

Plagiarism example 2

- Change variable names won't work

```
int j = count - 1;
//reversing the string by swapping
for (i = 0; i < count; i++)
{
  reverseNum[i] = number[j];
  j--;
}

  //Creating an integer array
  int arr[count];
  for (i = 0; i<count; i++) {
      arr[i] = reverseNum[i]-'0';
      printf("%d",arr[i]);
      }
  printf("\n");


  int sum1 = 0;
  //Getting the odd index sum
  for(int i=0;i<count;i=i+2)
  {


      sum1 = sum1 + arr[i];
  }
  int sum2 = 0;
  //Getting the even sum
  for(int i=1;i<count;i=i+2)
  {
      if (arr[i]*2<10){//For one digit product
      sum2+=arr[i]*2;
      }
      else {
          // separate the 2 digit answers by subtracting 10
          sum2 = sum2 + 1;


          sum2 = sum2 + (arr[i]*2)-10;
      }

  }
  printf("%d ", sum1);
  printf("%d\n",sum2);
  int totalSum = sum1 + sum2;
  // check if the sum of the partial sums is divisible by 0, thus valid
  if (totalSum%10==0){
  printf("valid\n");
  }
  else{
  printf("invalid\n");
  }



  return 0;
}
```

```
int j = count - 1;
//reversing the string by swapping
for (i = 0; i < count; i++)
{
  rev[i] = s[j];
  j--;
}

  int arr[count];

  // change the string to an integer array
  for (i = 0; i<count; i++) {
      arr[i] = rev[i]-'0';
      printf("%d",arr[i]);
      }
  printf("\n");


  int s1 = 0;
  // sum the digits at odd indexes
  for(int i=0;i<count;i=i+2)
  {


      s1 += arr[i];
  }
  int s2 = 0;
  // access the even index numbers
  for(int i=1;i<count;i=i+2)
  {
      if (arr[i]*2<10){
      s2+=arr[i]*2;
      }
      else {
          // separate the 2 digit answers by subtracting 10
          s2 += 1;


          s2 += (arr[i]*2)-10;
      }

  }
  printf("%d ", s1);
  printf("%d\n",s2);
  int finalsum = s1 + s2;
  // check if the sum of the partial sums is divisible by 0, thus valid
  if (finalsum%10==0){
  printf("valid");
  }
  else{
  printf("invalid");
  }



  return 0;
}
```

Plagiarism example 3

- Change comments won't work

```
Node * Addition(Node * n1, Node * n2) {
        bool AddOne = false;//to judge whether to add one in next loop, for example 300 +
        Node * current1 = n1, * current2 = n2; //set current to heads of n1 and n2
        int addend1 = 0, addend2 = 0, sum = 0;//change every current1 and current2 into in
        Node * head = NULL, * tail = NULL;

        while(current1 != NULL && current2 != NULL){
                addend1 = current1-> value;
                addend2 = current2-> value;
                if (AddOne) {
                        sum = addend1 + addend2 + 1;
                        if (sum > 999) {
                                AddOne = true;
                                sum -= 1000; //get the last three digits
                        }

                        else {
                                AddOne = false;
                        }
                }

                else {
                        sum = addend1 + addend2;
                        if (sum > 999) {
                                AddOne = true;
                                sum -= 1000;//get the last three digits
                        }
                        else {
                                AddOne = false;
                        }

                }

                Node * n = new Node;//build a forward linked list

            n->value = sum;
                n->next = NULL;

                if (head == NULL) {
                        head = n;
                        tail = n;
                }

                else {
                        tail->next = n;
                        tail = n;
                }

                current1 = current1->next;
                current2 = current2->next;
```

```
Node * Addition(Node * n1, Node * n2) {                         // Add n1 and n2
        bool PlusOne = false;                                                   // "plusor
        Node * current1 = n1, * current2 = n2;                  //
        int addend1 = 0, addend2 = 0, sum = 0;                  //
        Node * head = NULL, * tail = NULL;

        while(current1 != NULL && current2 != NULL){
                addend1 = current1-> value;
                addend2 = current2-> value;
                if (PlusOne) {                                  //
                        sum = addend1 + addend2 + 1;
                        if (sum > 999) {
                                PlusOne = true;
                                sum -= 1000;
                        }

                        else {
                                PlusOne = false;
                        }
                }

                else {
                        sum = addend1 + addend2;
                        if (sum > 999) {
                                PlusOne = true;
                                sum -= 1000;
                        }

                        else {
                                PlusOne = false;
                        }
                }

                Node * n = new Node;//build a forward linked list
            n->value = sum;
                n->next = NULL;

                if (head == NULL) {
                        tail = n;
                        head = n;
                }

                else {
                        tail->next = n;
                        tail = n;
                }

                current1 = current1->next;
                current2 = current2->next;


    }
```

Plagiarism example 4

- Change code style won't work

```
}

// Output the large number stored in the linked list
void print_num(Node *head) {

    Node * current = head;
    string numStr;

    while (current != NULL) {
        if (current != head) {
            string newStr = to_string(current->value);

            // Fill the zero if next is not NULL


            if (current->next != NULL)  {
                stringstream str;
                str << setw(3) << setfill('0') << newStr;
                str >> newStr;
            }
            numStr = newStr + numStr;
        }
        current = current->next;
    }
    cout << numStr << endl;
}

// Insert a value as a node to the head of a linked list
void head_insert(Node * & head, int v)
{
    Node * p = new Node;
    p->value = v;
```

```
// output the large number stored in the linked list
void print_num(Node *head)
{
    string str_num;
    Node *current = head;
    while (current != NULL)
    {
        if (current != head)
        {
            // convert int type to string
            stringstream ss;
            ss << current->value;
            string str_now_chunk = ss.str();

            // fill the zero of the value in current chunk


            if (current->next != NULL)
            {
                stringstream inter;
                inter << setw(3) << setfill('0') << str_now_chunk;
                inter >> str_now_chunk;
            }

            // concat values of all chunk
            str_num = str_now_chunk + str_num;
        }
        current = current->next;
    }
    cout << str_num << endl;
}
```

What do we expect from you?

- Be responsible for your learning progress
  - **It is expected that you spend roughly 10 hours per week for the course**
- Check Moodle, enable notification <- This is the only official direct communication channel for the course
- Follow course schedule
- **Read instructions carefully**
- Seek help whenever necessary
- Actively engaged in discussions (online/ offline)
- Be prepared to become a competent computer science professional

What kind of support do you have?

- Self-learning ≠ learn merely on your own without external help
- Self-learning = develop skills to acquire new knowledge (including skills to seek help and advice)

Seek help and support form:

- Moodle online Q&A and discussions
- Problem-solving sessions
- Consultations

Tips for getting effective help

What **NOT** to do when seeking help

- Ask without doing some research if there are already answers to similar issues

Be well-prepared before you ask

- Merely throw your codes at TAs / STAs and hoping that they will debug for you
- Post your assignment code or answers to Moodle

Tips for getting effective help

- Highly recommended to have online communications on Moodle so everyone can share and learn
- How to ask?
  - Be specific

Bad example
I don't understand.

Good example
Why isn't this variable effective inside this loop?

  - Provide the context / background

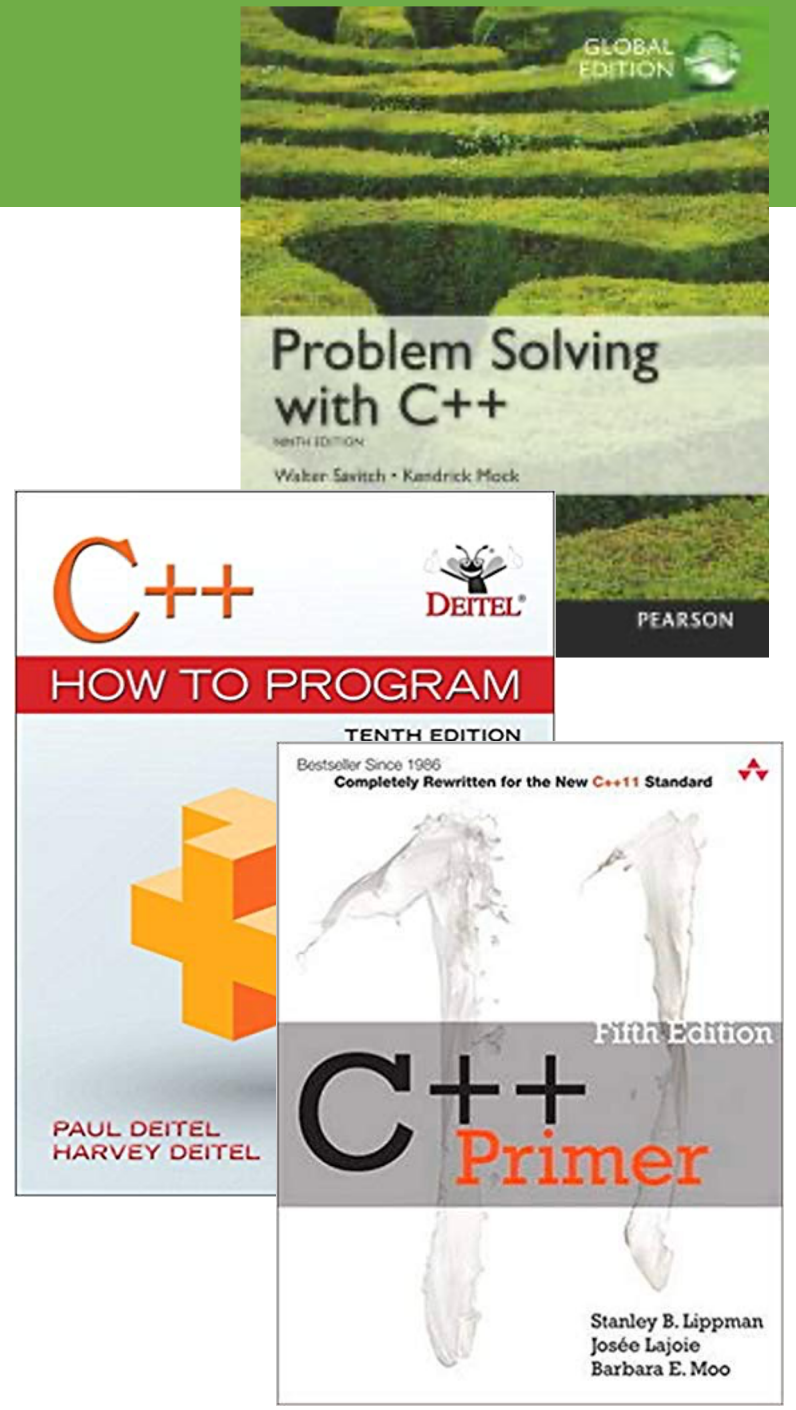
Bad example
It doesn't work.
Why?

Good example
I got this error when compiling my code with this Makefile

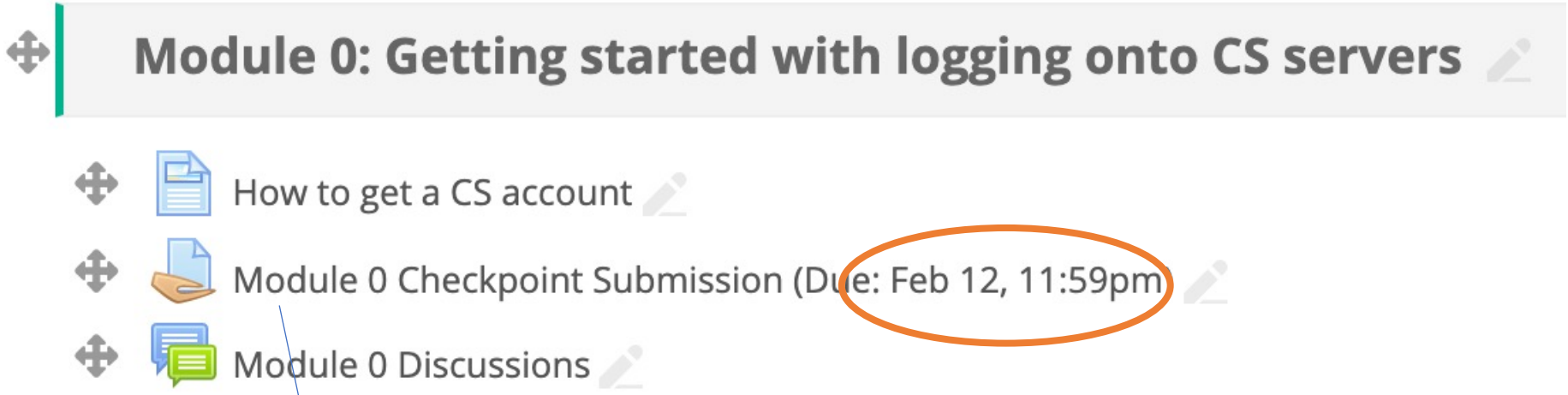
How do I ask a good question? (stackoverflow)
⇨ https://stackoverflow.com/help/how-to-ask

References

- You don't need these books unless you do
  - Problem Solving with C++
  - C++ How to Program
  - C++ Primer
  - A point to note: we will teach C++ as well as C
    - Strings (C-string vs. string class)
    - Standard I/O (printf/scanf vs. cin/cout)
    - file I/O (FILE * vs. fstream)
    - Memory management (malloc/free vs new/delete)



Get Started Now



Module 0 Checkpoint Submission (Due: Feb 12, 11:59pm)

Your first task: Finish Module 0 (available already) and Module 1 (available by early next week) checkpoints and submit by the deadline in the third week