C/C++ workshop

Practice

|  |
| --- |
| Notes |
| * Each question has 100 points (100%). Score of each question is depend on how many percent this question has been completed. * Score of sub-questions in a question are the same, and counted by 100/number-of-sub-questions. * Score of practice is average score of all questions * Candidate zip all answers (doc file and zipped-project file in <your\_name>.zip |

**Question 1: Hello world**

Write your answer in word document (includes code) in Q1\_<yourname>.doc

|  |
| --- |
| Q 1.1 |
| Use Visual Studio, code a simple program, display “hello C/C++” in console screen |

|  |
| --- |
| Q 1.2 |
| Try to find-out, where is .obj file and .exe are generated |

|  |
| --- |
| Q 1.3 |
| Open project property. Try to explain some elements in tab General: Output Directory, Intermediate Directory, Extensions to Delete on Clean, Build Log File, and Configuration Type. (use google) |

|  |
| --- |
| Q 1.4 |
| Modified entry point like:   |  | | --- | | #include <stdio.h>  #include <iostream>  int main(int n, char\*\* args)  {  printf("Number of Param %d\n", n);  for (int i = 0; i < n; i++)  {  printf("param[%d] = %s\n", i, args[i]);  }  system("pause");  } |   Try to run, and watch the result in console screen and explain. |

|  |
| --- |
| Q 1.5 |
| Use the code in Q1.4. Open project property, and select debugging tab. Type “1 2 3 4” into Command Arguments    Try to run, and watch the result in console screen and explain. |

|  |
| --- |
| Q 1.6 |
| Code a program: make an infinity loop, limit frame-rate to 5. Each frame, print to console screen current time.  HINT:   * use clock(), time() of time.h; * use sleep() of windows.h |

**Question 2: Variable**

Write your answer in word document (includes code) in Q2\_<yourname>.doc

|  |
| --- |
| Q 2.1 |
| Use a nest 3-for-loops, calculate the time to run when use, and not use register. Test in several times, and give a statistics when MAX is 10, 100, 1000, 10000   |  |  | | --- | --- | | for (int i = 0; i < MAX; i++)  {  for (int j = 0; j < MAX; j++)  {  for (int k = 0; k < MAX; k++)  {  }  }  } | for (register int i = 0; i < MAX; i++)  {  for (register int j = 0; j < MAX; j++)  {  for (register int k = 0; k < MAX; k++)  {  }  }  } |  |  |  |  | | --- | --- | --- | |  | Non-register | Register | | 10 |  |  | | 100 |  |  | | 1000 |  |  | | 10000 |  |  | |

|  |
| --- |
| Q 2.2 |
| Implement the following   |  | | --- | | void calculate(int val)  {  static int count=10;  static int value;  printf("%d %d\n",value, count++);  value = val;  }  int main(int n, char\*\* args)  {  int i, j;  calculate(i);  calculate(j);  } |   Look at the result and explain why? |

|  |
| --- |
| Q 2.3 |
| For counting number of times a function has been called, we use three ways:   * Global variable * Static variable inside function * Static variable outside function   Implements those ways, and compare |

|  |
| --- |
| Q 2.4 |
| A programmer codes his program with 2 files: main.cpp and another.cpp. In another.cpp, he declares a **global** **variable** named myVar with initial value is 10. In main .cpp, he needs to print value of myVar into screen. And he tries the following ways:   * Declare myVar in another.cpp, and in print it in main() in main.cpp * Declare myVar in another.cpp. Declare again myVar in main.cpp and print it * Declare myVar in another.cpp. In main.cpp, declare myVar with extern keyword, and print it.   He’s got problem with the first and second way. The third is run well.  Please explain what happen with the first and second; explain the reason why the third is OK and try to implement it. |

|  |
| --- |
| Q 2.5 |
| Detect & solve problem of the following code:   |  | | --- | | #include <stdio.h>  const int xyz;  int main(int n, char\*\* args)  {  xyz = 0;  printf("%d", xyz);  } | |

**Question 3: Primary data type**

Write your answer in word document (includes code) in Q3\_<yourname>.doc

|  |
| --- |
| Q 3.1 (\*) |
| * Run the following code, and explain the result. * If replacing “char c” with “unsigned char c”, what will happen, and why? * Set “i” value to 260. Explain the result. * Supposing that “i” is always from 0 to 255. Try to give a solution to get j’s value equal to i’s value. (does not used unsigned char)  |  | | --- | | #include <stdio.h>  #include <iostream>  int main(int n, char\*\* args)  {  int i = 140;  char c = i;  int j = c;  printf("%d", j);  system("pause");  } | |

|  |
| --- |
| Q 3.2 |
| Run the following code, and explain the result.   |  | | --- | | #include <stdio.h>  #include <iostream>  int main(int n, char\*\* args)  {  int i = 5;  while (i = 5)  {  printf("Inside loop\n");  i--;  }  } | |

|  |
| --- |
| Q 3.3 |
| Implement functions:   * int Floor(float x): Returns the next lowest integer value by rounding down value if necessary. For example: * floor(6.3) = 6 * floor(6.9) = 6 * int ceil(float x): Returns the next highest integer value by rounding down value if necessary. For example: * ceil(6.3) = 7 * ceil(6.9) = 7 * int round(float x): return the round value. * round(6.3) = 6 * round(6.9) = 7 |

**Question 4: Array**

Write your answer in word document (includes code) in Q4\_<yourname>.doc

|  |
| --- |
| Q 4.1 |
| Write program to ask the user to type 10 integers of an array. The program must compute and write the number of integers greater or equal to 10. |

|  |
| --- |
| Q 4.2 |
| Write a program which takes 2 arrays of 10 integers each, a and b. c is an array with 20 integers. The program should put into c the appending of b to a, the first 10 integers of c from array a, the latter 10 from b. Then the program should display c.   * Use memcpy * Without memcpy |

|  |
| --- |
| Q 4.3 (\*) |
| Explain the result:   |  | | --- | | #include <stdio.h>  #include <iostream>  int main(int n, char\*\* args)  {  int a[10];  for (int i = 0; i < 10; i++)  {  a[i] = i\*10;  }  printf("0x%x 0x%x 0x%x\n", a, \*a, &a);  printf("0x%x 0x%x 0x%x 0x%x\n", (a + 1), \*(a + 1), \*a + 1, &a + 1);  } | |

|  |
| --- |
| Q 4.4 |
| Implement bubble sort algorithm (sort acceding) to sort an array from “offset” with “count” elements.  For example: a[5] = {100, 4, 2, 6, 3} sort(a, 1, 4) {100, 2, 3, 4, 6}  void BubbleSort(int arr[], int offset, int count) |

|  |
| --- |
| Q 4.5 (\*) |
| Implement functions to invert member of an array   * void Invert(int input[], int num\_element): invert input array. * void Invert(int input[], int num\_element, int output[]): generate output array from inverted input. Note: after executing this function, input array is not changed |

|  |
| --- |
| Q 4.6 |
| Explain the result:   |  | | --- | | #include <iostream>  using namespace std;  int x = 11;  int y = 12;  int a[5] = {21, 22, 23, 24, 25};  int u = 31;  int v = 32;  void main()  {  for (int i = -2; i < 7; i++)  cout << i << " " << a[i] << endl;  } | |

**Question 5: Pointer**

Write your answer in word document (includes code) in Q5\_<yourname>.doc

|  |
| --- |
| Q 5.1 (\*) |
| Explain the result   |  | | --- | | #include <stdio.h>  #include <iostream>  using namespace std;  typedef int \*IntPtrType;  int main()  {  IntPtrType ptr\_a, ptr\_b, \*ptr\_c;  ptr\_a = new int;  \*ptr\_a = 3;  ptr\_b = ptr\_a;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  ptr\_b = new int;  \*ptr\_b = 9;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  \*ptr\_b = \*ptr\_a;  cout << \*ptr\_a << " " << \*ptr\_b << "\n";  delete ptr\_a;  ptr\_a = ptr\_b;  cout << \*ptr\_a << " " << \*&\*&\*&\*&\*ptr\_b << "\n";  ptr\_c = &ptr\_a;  cout << \*ptr\_c << " " << \*\*ptr\_c << "\n";  delete ptr\_a;  ptr\_a = NULL;  return 0;  } | |

|  |
| --- |
| Q 5.2 |
| Introduce int variables x and y and int\* pointer variables p and q. Set x to 2, y to 8, p to the address of x, and q to the address of y. Then print the following information:   * The address of x and the value of x. * The value of p and the value of \*p. * The address of y and the value of y. * The value of q and the value of \*q. * The address of p (not its contents!). * The address of q (not its contents!).   Use the Hex function to print all pointer/address values and format the output so it is easy to make comparisons. |

|  |
| --- |
| Q 5.3 |
| Introduce int variables x, y, z and int\* pointer variables p, q, r. Set x, y, z to three distinct values. Set p, q, r to the addresses of x, y, z respectively.   * Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r. * Print the message: Swapping values. * Execute the swap code: z = x; x = y; y = z; * Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.   Draw diagrams to explain the results |

|  |
| --- |
| Q 5.4 |
| Introduce int variables x, y, z and int\* pointer variables p, q, r. Set x, y, z to three distinct values. Set p, q, r to the addresses of x, y, z respectively.   * Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r. * Print the message: Swapping pointers. * Execute the swap code: r = p; p = q; q = r; * Print with labels the values of x, y, z, p, q, r, \*p, \*q, \*r.   Draw diagrams to explain the results |

|  |
| --- |
| Q 5.5 (\*) |
| Draw diagrams to explain the results:   |  | | --- | | #include <iostream>  using namespace std;  void main()  {  int a[4] = {1, 2, 3, 4};  int \*p = a;  printf("0x%x 0x%x 0x%x\n", a, \*a, &a);  printf("0x%x 0x%x 0x%x\n", p, \*p, &p);  printf("0x%x 0x%x 0x%x 0x%x\n", (a + 1), \*(a + 1), \*a + 1, &a + 1);  printf("0x%x 0x%x 0x%x 0x%x\n", (p + 1), \*(p + 1), \*p + 1, &p + 1);  } | |

|  |
| --- |
| Q 5.6 (\*) |
| Detect and solve problems of following program:   |  | | --- | | void main()  {  int a[4] = {1, 2, 3, 4};  int \*p = a;  int \*p2 = new int;    delete p;  delete a;  delete p2;  } | |

|  |
| --- |
| Q 5.7 (\*) |
| * Why should we use delete? * When we use delete? * Difference between delete and delete[]. Write a demo |

|  |
| --- |
| Q 5.8 (\*) |
| Detect and solve problems of following program:   |  | | --- | | #include <iostream>  using namespace std;  #define COUNT 10  #define MAX(a, b) ((a) < (b) ? (a):(b))  void main()  {  int \*p = new int[COUNT];  int a[COUNT];  for (int i = 0; i < sizeof(a); i++)  {  a[i] = i;  }  for (int i = 0; i <; i++)  {  p[i] = i;  }    for (int i = 0; i < MAX(sizeof(a), sizeof(p)); i++)  {  printf("%d %d\n", a[i], p[i]);  }  } | |

|  |
| --- |
| Q 5.9 (\*) |
| Explain the result   |  | | --- | | void main()  {  int \*a = new int;  \*a = 10;  delete a;  printf("%d", \*a);  } | |

**Question 6: String**

Write your answer in word document (includes code) in Q6\_<yourname>.doc

|  |
| --- |
| Q 6.1 |
| Print a string “Hello world” to screen, by using:   * char \*st = “Hello world” * char st[] = {‘H’, ‘e’, ….} * char \*st = new char[…] …. |

|  |
| --- |
| Q 6.2 |
| Implement a function to clone a string:   * void clone(char \*input, char \*output) |

|  |
| --- |
| Q 6.3 (\*\*) |
| Implement a function to copy a string:  void copyString(char \*input, int offset, int length, bool invert, char \*output, int output\_offset)   * input: input string * offset: starting position for copy * length: length of substring to be copied * invert: invert the result * output: output string * output\_offset: the beginning of copied position of output   for example:  st = “Hello world”  st2 = “My name is C++”  copyString(st, 6, 5, true, st2, 11) st2 = “My name is dlr”;  Implement this function in 3 ways:   * using memcpy * using strcpy * without memcpy and strcpy |

**Question 7: Enum & Union**

Write your answer in word document (includes code) in Q7\_<yourname>.doc

|  |
| --- |
| Q 7.1 |
| Explain the result   |  | | --- | | #include <iostream>  using namespace std;  enum SEASON {SPRING = 0x1, SUMMER = 0x10, FALL = 0x100, WINTER};  void CheckSeason(int season)  {  switch (season)  {  case SPRING:  printf("SPRING is %d\n", SPRING);  break;  case SUMMER:  printf("SUMMER is %d\n", SUMMER);  break;  case FALL:  printf("FALL is %d\n", FALL);  break;  case WINTER:  printf("WINTER is %d\n", WINTER);  break;  default:  printf("UNKNOWN: %d", season);  break;  }  }  void main()  {  SEASON season = WINTER;  CheckSeason(season);  CheckSeason(1);  CheckSeason(5);  } | |

|  |
| --- |
| Q 7.2 |
| Alignment is divided into 2 group: vertical alignment and horizontal alignment. For horizontal alignment, we have values: LEFT, CENTER, and RIGHT. For vertical alignment, we have: TOP, CENTER, and BOTTOM.   * Try to define two enum Vertical\_Align and Horizontal\_Align with their above values * Detect and solve problems when define enums. |

|  |
| --- |
| Q 7.3 |
| Draw diagrams to explain the results:   |  | | --- | | union MyUnion  {  int iVal;  float fVal;  char cVal[5];  };  void main()  {  union MyUnion u;  u.fVal = 100;  printf("%d\n%f\n%d\n%d\n", u.iVal, u.fVal, u.cVal[0], sizeof(u));  } | |

**Question 8: Struct**

Use visual studio, named your project as Q8\_<yourname>. Zip and send your project

|  |
| --- |
| Q 8.1 |
| Date information includes:   * Day of week : 0 for Sunday and 6 for Saturday (unsigned char) * Day: 0..31 (unsigned char) * Month: 0..12 (unsigned char) * Year: xxxx (unsigned int)   Implement:   * Define struct of date * Get date of system and save to a struct instance * Print struct information as format: * dow, dd-mm-yyyy * dd/mm/yyyy * mm-dd-yyyy * dow dd.mm.yy |

|  |
| --- |
| Q 8.2 (\*) |
| Use struct of 8.1. Implement:   * Print size of struct * Declare year in second (dow, year, day, month), and print size of struct. Draw diagrams to explain the result * Build project with 1 byte alignment, print size of struct in two above cases and explain the result. |

|  |
| --- |
| Q 8.3 (\*\*) |
| Use struct of 8.1. Implement:   * Constructor with dow, day, month, year. * Implement function check valid date (valid of dow, day, month year) |

|  |
| --- |
| Q 8.4 (\*) |
| Student information include   * name (char \*) * id (int) * score (float) (0 to 10)   Implement:   * Struct of student * A class room is included ten students. (List of student is stored in an array). * Implement a function to input the student information (check if duplicated student ID, and value of score must be in range) * Sort list of student by score. |

|  |
| --- |
| Q 8.5 (\*\*) |
| Use visual studio, create new project. Named your project as Q8\_5\_<yourname>. Zip and send your project.  Same as Q8.4, however, students are stored in a linked-list structure:   |  | | --- | | typedef struct \_Student  {  char \*Name;  int ID;  float Score;  struct \_Student \*next;  } Student; |  * Input student information until ID is equal to zero (check if duplicated student ID, and value of score must be in range) * Sort list by score * Remove all students get score less than 5 |

**Question 9: Function**

Write your answer in word document (includes code). in Q9\_<yourname>.doc

|  |
| --- |
| Q 9.1 (\*) |
| Implement a function like printf:  void show(char \* const format, …)   * Format : * %ud: unsigned integer * %sd: signed integer |

|  |
| --- |
| Q 9.2 |
| Implement a function function SayHello, to print screen “Hello world”   * No parameter * A string, print to screen “hello world” and that string * A string, and a bool: If bool value is true, convert to string to capital; print to screen “hello world” and modified string. (bool parameter has default value is false). |

|  |
| --- |
| Q 9.3 (\*) |
| Implement a sort function (bubble sort) with unknown order preference (ascending or descending). The order preference is decided by a pointer function parameter.  void sort(int \*arr, int count, int (\*order) (int, int));  int CheckBigger(int a, int b) {return a>b;}  int CheckSmaller(int a, int b) {return a<b;} |

|  |
| --- |
| Q 9.4 (\*\*) |
| Use visual studio, create new project. Named your project as Q10\_4\_<yourname>. Zip and send your project.  Same requirement as Q9.3, however, sort function is implemented in another file:   * “**sort\_cpp.cpp**” (use extern) * “**sort\_c.c**” (use extern “C”) |

|  |  |
| --- | --- |
| Q 9.5 (\*) |  |
| Introduce classA and several functions. Draw diagrams to explain the results:   |  |  | | --- | --- | | class **ClassA**  {  public:  ClassA(int val = 0) {m\_pVal = val;}  int m\_pVal;  }; | | | void **Todo1**(ClassA \*A, ClassA \*B)  {  ClassA \*tmp = A;  A = B;  B = tmp;  } | void **Todo2**(ClassA &A, ClassA &B)  {  ClassA tmp = A;  A = B;  B = tmp;  } | | void **Todo3**(ClassA A, ClassA B)  {  ClassA tmp = A;  A = B;  B = tmp;  } | void **Todo4**(ClassA \*&A, ClassA \*&B)  {  ClassA \*tmp = A;  A = B;  B = tmp;  } |  |  | | --- | | void main()  {  ClassA \*A = new ClassA(10);  ClassA B(0);  ClassA \*C = &B;  printf("%d %d %d\n", A->m\_pVal, B.m\_pVal, C->m\_pVal);  **Todo1**(A, &B);  printf("%d %d %d\n", A->m\_pVal, B.m\_pVal, C->m\_pVal);  **Todo2**(\*A, B);  printf("%d %d %d\n", A->m\_pVal, B.m\_pVal, C->m\_pVal);  **Todo3**(\*A, B);  printf("%d %d %d\n", A->m\_pVal, B.m\_pVal, C->m\_pVal);  **Todo4**(A, C);  printf("%d %d %d\n", A->m\_pVal, B.m\_pVal, C->m\_pVal);  } | | |

**Question 10: Class**

Use visual studio, create new project. Named your project as Q10\_<yourname>. Zip and send your project.

|  |
| --- |
| Q 10.1 |
| Introduce class for people:   * Properties * [protected] Name: char\* * [protected] Birthday: struct Date * [protected] ID code: int * Methods: * [public - virtual] PrintInfo: display information * [public] Get/set name, birthday, ID code   Introduce class for staff: (inherit from people)   * Properties * [protected] Department ID: int * [protected] Joined Date: structDate * [protected] Level: int * Methods: * [public - virtual] PrintInfo: display information (+ people information) * [public] Get/set department ID, joined date, level   Introduce class for student: (inherit from people)   * Properties * [private] Grade: int * [private] Number of project: int * [private] Score of each project: float\* * Methods: * [public] PrintInfo: display student information (+ people information) * [public] Get/set grade, number of project, score * [private] Calculate average score * [public] Get average score   Introduce class for teacher: (inherit from people and staff)   * Properties * [private] Class: int * [private] Subject: enum {MATHS, PHYS, ENGLISH} * [private] Second subject: enum {MATHS, PHYS, ENGLISH} * Methods: * [public] PrintInfo: display teacher information (+ people and staff information) * [public] Get/set class, subject, second subject   All people are stored in a linked-list:   |  | | --- | | class Node  {  public:  People\* data;  Node\* next;  }; |   Implement above introduction. |

|  |
| --- |
| Q 10.2 |
| Implements:   * Implements counting number of peoples, staffs, teachers, students are created (use static member) * In node, implement static function to add element * Add to list 5 staff, 5 teacher, and 5 student * Sort list group by type: staff first, teacher next, and student last * Print information of all students have had average score more than 5.0 |

|  |
| --- |
| Q 10.3 (\*) |
| Implements:   * Implement copy constructor * Clone current list, using copy constructor for each element |

|  |
| --- |
| Q 10.4 (\*) |
| Implements:   * Implement static methods remove element for node. Make sure class memory was clean up. (When delete student, memory of people MUST BE clean too – use virtual destructor and println to check when an instance is destroyed). |

|  |
| --- |
| Q 10.5 (\*\*) |
| Implements:   * Save a list to of student to file * Load list of student from file |

**Question 11: C/C++ advance features**

|  |
| --- |
| Q 11.1 |
| Write your answer in word document (includes code) in Q11\_1\_<yourname>.doc  Define a template function to calculate average values of an array:  template <class R, class T> R Average(int count, T\* list) |

|  |
| --- |
| Q 11.2 |
| Use visual studio, create new project. Named your project as Q11\_2\_<yourname>. Zip and send your project.  Use 10.1, define a template class node   |  | | --- | | template <class T>  class Node  {  public:  T\* data;  Node\* next;  }; |   Implement methods:   * Add a node * Remove a node * Remove all nodes |

|  |
| --- |
| Q 11.3 |
| Use visual studio, create new project. Named your project as Q11\_3\_<yourname>. Zip and send your project. Use 10.1.   * Supposing that we have an instance of teacher, try to cast to * Staff * People * Supposing that we have an instance of student, try to cast to * People * Teach   By using explicit casting, dynamic/static casting.  (Test by call all member methods after casting) |

|  |
| --- |
| Q 11.4 |
| Use visual studio, create new project. Named your project as Q11\_4\_<yourname>. Zip and send your project. Implement   * Divide function, support division by zero exception * Calculate average of a list, support out of bound exception, division by zero exception: * float Average(int count, int offset, float \*arr) |

**Question 12: STL**

Use visual studio, create new project. Named your project as Q12\_<yourname>. Zip and send your project.

|  |
| --- |
| Q 12.1 |
| Implement 10.1, but use STL list instead of Node structure.  Support:   * Add element (people, staff, student, teacher) * Modified people, staff, student, teacher information * Remove element, check memory clean up * Use VLD to detect leak |

|  |
| --- |
| Q 12.2 |
| Compile 12.1 by using GNU GCC through Cygwin. Attach your makefile into Q12\_<yourname> package |

**Question 13: Convert from Java to C**

Use visual studio, create new project. Named your project as Q13\_<yourname>. Zip and send your project.

|  |
| --- |
| Q 13.1 |
| Convert the following code from Java to C++  Note: Object in Java is same as void\* in C++   |  | | --- | | /\* class Stack  \*/  public class Stack  **{**  private Object**[]** stack **;**  private int total**;** // to track number of items  public Stack**(**int size**)**  **{**  stack **=** **new** Object**[**size**];** // create array  total **=** 0**;** // set number of items to zero  **}**  /\*\*  \* add an item to the array  \*/  public boolean push**(**Object obj**)**  **{**  **if** **(** isFull**()** **==** **false)** // checks if space in stack  **{**  stack**[**total**]** **=** obj**;** // add item  total**++;** // increment item counter  **return** **true;** // to indicate success  **}**  **else**  **{**  **return** **false;** // to indicate failure  **}**  **}**  /\*\*  \* remove an item by obeying LIFO rule  \*/  public Object pop**()**  **{**  **if** **(**isEmpty**()** **==** **false)** // check stack is not empty  **{**  // reduce counter by one  Object obj **=** stack**[**total**-**1**];** // get last item  stack**[**total**-**1**]=** **null;** // remove item from array  total**--;** // update total  **return** obj**;** // return item  **}**  **else**  **{**  **return** **null;** // to indicate failure  **}**  **}**  /\*  \* checks if array is empty  \*/  public boolean isEmpty**()**  **{**  **if** **(**total **==**0**)** **{**  **return** **true;**  **}**  **else**  **{**  **return** **false;**  **}**  **}**  /\*\*  \* checks if array is full  \*/  public boolean isFull**()**  **{**  **if** **(**total **==**stack**.**length**)**  **{**  **return** **true;**  **}**  **else**  **{**  **return** **false;**  **}**  **}**  /\*\*  \* returns the item at index i  \*/  public Object getItem**(**int i**)**  **{**  **return** stack**[**i**-**1**];** // ith item at position i-1  **}**  /\*\*  \* return the number of items in the array  \*/  public int getTotal**()**  **{**  **return** total**;**  **}**  **}** | |

|  |
| --- |
| Q 13.2 |
| Use 13.1, write an C/C++ demo with stack of integer |