

EXAMINATIONS – 2019

TRIMESTER 1

NWEN 241
SYSTEMS PROGRAMMING

Time Allowed: TWO HOURS

CLOSED BOOK

Permitted materials: Only silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination.

No electronic dictionaries are allowed.

Paper foreign to English language dictionaries are allowed.

Instructions: Attempt ALL TEN (10) questions:

1. C/C++ Fundamentals. [15 marks]
2. User-Defined Types and C++ Classes. [15 marks]
3. Arrays. [10 marks]
4. Pointers. [10 marks]
5. Dynamic Memory Allocation. [15 marks]
6. C++ Templates and Vectors. [10 marks]
7. Data Structures. [10 marks]
8. File I/O. [10 marks]
9. Low-Level and Socket Programming. [15 marks]
10. Process Management. [10 marks]

The examination consists of 120 marks in total.

1. C/C++ Fundamentals. (15 marks)

- (a) Declare a constant MYCONST with value 1024 using appropriate preprocessor directive. (2 marks)

```
#define MYCONST 1024
```

- (b) What value does the C++ expression `float(5 / 2)` evaluate to? (2 marks)

The float 2.0

- (c) Consider the following C++ code snippet: (2 marks)

```
namespace ns
{
    int a = 100;
    void incr(void)
    {
        a++;
    }
}
```

Write a **single line of code** to invoke the function `incr()` from outside the namespace `ns`.

```
ns::incr();
```

- (d) Consider the following C/C++ program: (2 marks)

```
#include <stdio.h>

int macro_me(int a, int b)
{
    return a+++b;
}

int main(void)
{
    int i = 7;
    int j = macro_me(1+2, i);
    printf("%d,%d", i, j);
    return 0;
}
```

What is the output of the program?

7,24

- (e) Re-write `macro_me(int a, int b)` in program in (d) into a function like macro `FLM(A, B)`, such that when the call to `macro_me(1+2, i)` in the program is replaced with `FLM(1+2, i)`, the outputs will remain the same. **(3 marks)**

```
#define FLM(A,B) ((A)*(B)+1)
```

- (f) Consider the following C++ program: **(4 marks)**


```
#include <iostream>

int main(void)
{
    char c = 'A';
    int i = 10;
    float f = 2.5;

    std::cout << c << ", " << i << ", " << f;
```

```
    return 0;  
}
```

Re-write the program to use only functions from `stdio.h`.



```
#include <stdio.h>  
  
int main(void)  
{  
    char c = 'A';  
    int i = 10;  
    float f = 2.5;  
  
    printf("%c,%d,%f", c, i, f);  
  
    return 0;  
}
```

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

2. User-Defined Types and C++ Classes. (15 marks)

- (a) Define a structure that can represent the coordinates of a point in two dimensions, with tag `coord` and consisting of 2 `float` members `x` and `y`. (3 marks)

```
struct coord { float x; float y; };
```

- (b) Use `typedef` to define a new type `coord_t` from the structure defined in (a). (2 marks)

```
typedef struct coord coord_t;
```

- (c) Define an enumeration type with identifiers `quad`, `penta`, and `hexa` having values of 4, 5, and 6, respectively. Use `pref` as tag of the enumeration type. (3 marks)

```
enum pref { quad = 4, penta, hexa };
```

- (d) Declare a variable `p` of type defined in (c) and with initial value `hexa`. (2 marks)

```
enum pref p = hexa;
```

(e) Consider the following C++ class declaration:

(5 marks)

```
namespace nsA {
    class ClassA {
    public:
        virtual int f1() const = 0;
        virtual int f2(void) = 0;
    protected:
        int a;
    };
}
```

Declare a class `ClassB` that extends `ClassA` but in a different namespace called `nsB`. `ClassB` should preserve the access specifier of the members, should not be abstract, and should have an inline default constructor that initializes the member variable `a` to 100.

(Hint: You do not need to show function implementations, just the prototype declarations)

```
namespace nsB {
    class ClassB: public nsA::ClassA {
    public:
        int f1() const;
        int f2();
        ClassB() : a(100) {}
    };
}
```

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

3. Arrays. (10 marks)

Given the following array and pointer declarations

```
int ia[] = {1,2,3,4,5,6,7,8,9};  
int *iap = &ia[0];
```

- (a) Write 3 C expressions showing 3 different ways to access the value stored in the first element of the array `ia`. **(3 marks)**

```
ia[0]  
*ia  
*iap
```

- (b) Suppose that the base address of the array `ia` is at (decimal) 1000. Supposing that an `int` occupies 32 bits, what is the value of `iap + 2`? **(2 marks)**

```
1000 + 2*4 = 1008
```

- (c) Write a for-loop to iterate through the array outputting each element using array indexes. You may use either `printf()` or `cout` to display the element. **(3 marks)**

```
for (int i = 0; i<9;i++)  
    printf("%d ", ia[i]); // cout << ia[i];
```

- (d) Write a for-loop to iterate through the array displaying each element using pointers. You may use either `printf()` or `cout` to display the element. **(2 marks)**

```
for (iap = ia; iap < ia + 9; iap++)  
    printf("%d", *iap); // cout << *iap;
```

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

4. Pointers. (10 marks)

Given the following variable declarations

```
int j, k;  
double m;  
int *p1, *p2;
```

- (a) Write a statement to assign the address of j to p1.
- (2 marks)**

```
p1 = &j;
```

- (b) Declare another pointer p3 that can hold a pointer to any data type.
- (2 marks)**

```
void *p3;
```

- (c) Write a statement to assign the pointer address in p1 to p2.
- (2 marks)**

```
p2 = p1;
```

- (d) Write a statement to assign the address of m to p3.
- (2 marks)**

```
p3 = &m;
```

- (e) Write a statement to assign the value of whatever p3 points to k.
- (2 marks)**

```
k = *(double*) p3; //or k = (int)*(double*) p3;
```

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

5. Dynamic Memory Allocation. (15 marks)

- (a) Describe briefly 3 common issues encountered when using dynamic memory. (3 marks)

Possible answers:

- Returning a pointer to an automatic variable
- Heap block overrun - similar to array going out of bounds
- Memory leak loss of pointer to allocated memory
- Freeing non-heap or unallocated memory
- Not freeing dynamic memory
- Not checking to ensure the memory allocation operation was successful (for NULL)

- (b) Describe one difference between `calloc` and `malloc` in terms of how they initialize the contents of the newly allocated memory. (2 marks)

`calloc` is clear and allocate the memory therefore sets the memory to 0, whereas `malloc` does not clear the memory.

- (c) Why are `new` and `delete` the preferred method of managing dynamic memory in C++? (2 marks)

new and delete call the constructors and destructors of objects, whereas malloc, calloc, realloc and free do not.

new is typesafe and throws an error if it fails, and can be overloaded.

(d) Given the following program:

```
#include <stdio.h>

void strange (int x)
{
    static int y;

    if ( x == 0 )
        printf( "%d\n", y );
    else if ( x == 1 )
        y = 25;
    else if ( x == 2 )
        y++;
}

int main (void)
{
    strange(1); //first function call
    strange(0); //second function call
    strange(2); //third function call
    strange(0); //fourth function call

    return 0;
}
```

i. What is the initial value of y? **(2 marks)**

0

ii. What is the value of y after the first call to function strange? **(2 marks)**

25

iii. What is the value of `y` after the fourth call to function `strange`? **(2 marks)**

26

iv. What is the output of the program? **(2 marks)**

25

26

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

6. C++ Templates and Vectors. (10 marks)

(a) In C++, what does the Standard Template Library (STL) define? (3 marks)

Containers, Algorithms, Iterators

(b) Generic function templates are used to define functions for what data types? (2 marks)

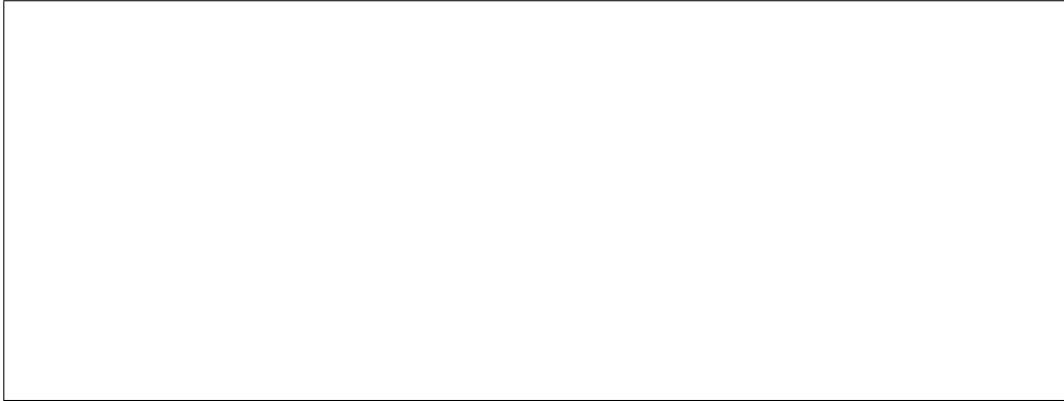
Types can be basic, derived and user defined data types (classes / structures)

(c) Write a generic function to return the minimum of two parameters/arguments. (2 marks)

```
template<typename T> // or template<class T>
T min(T a, T b) {
    return a < b ? a : b;
}
```

3 marks

(d) Give 3 advantages of using the generic vector class over a C-style array. (3 marks)



Possible answers:

- There is a single definition of the vector container, but it can be used to define many different kinds (data types) of vectors.
- Vectors are similar to dynamic arrays with the ability to resize automatically when an element is inserted or deleted.
- Their storage is handled automatically by the container.
- Vectors also have safety features that make them easier to use than arrays, automated bounds checking and memory management.
- Vector elements are placed in contiguous storage so that they can be accessed and traversed using iterators.

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

7. Data Structures. (10 marks)

(a) Briefly describe one advantage of linked lists over arrays. (2 marks)

Possible answers:

- The size of the arrays is fixed: So we must know the upper limit on the number of elements in advance.
- In general, the allocated memory is equal to the upper limit irrespective of the usage.
- Inserting a new element in an array of elements is expensive, because room has to be created for the new elements and to create room existing elements have to be shifted.
- Deleting an element in an array of elements is expensive, because elements have to be shifted to avoid unused gaps in between elements.

(b) In C, a node in a list is implemented using a structure. Declare a C structure with tag `node` that defines a node of a *doubly* linked list. For simplicity, declare the data field to be of type `int` with identifier `data`. (2 marks)

```
struct node
{
    int data;
    struct node *next; // pointer to next element
    struct node *prev; // pointer to previous element
};
```

- (c) In C++, STL has container classes to implement two types of list. What are the names of these container classes? **(2 marks)**

Forward list and list

- (d) What is the output of this C++ program? **(4 marks)**

```
#include <iostream>
#include <list>
#include <iterator>

using namespace std;

// Print the elements in a list
void showlist(list <int> l)
{
    list <int> :: iterator it;
    for(it = l.begin(); it != l.end(); ++it)
        cout << *it << ' ';
    cout << '\n';
}

int main()
{
    list <int> list1;
    for (int i = 1; i < 10; ++i) {
        list1.push_back(i);
    }
    list1.pop_front();
    list1.reverse();
    showlist(list1);
    return 0;
}
```

Student ID:

98765432

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

8. File I/O. (10 marks)

- (a) In C++, the `iostream` and `fstream` header files are typically used for file input and output. Name the 3 classes used for declaring file streams in C++. **(3 marks)**

`ofstream, ifstream, andfstream`

- (b) Write a C++ code that will declare and open a binary file `picture.gif` for input. **(1 mark)**

```
ifstream ifs;  
ifs.open ("picture.gif", ios::binary);
```

- (c) In C++, what is the command to clear an output stream buffer? **(1 mark)**

`ostream::flush`

- (d) Write a C statement that will open a binary file `output.bin` for output. **(3 marks)**

```
FILE *fp = fopen("output.bin", "wb");
```

(e) Consider the following C program:

(2 marks)

```
#include <stdio.h>

int main()
{
    char c;
    FILE *infp = fopen("infile.txt", "r");
    FILE *outfp = fopen("outfile.txt", "w");

    while( (c=getc(infp)) != EOF ) {
        if (c != ' ') {
            putchar(c+1, outfp);
        }
    }
    fclose(infp);
    fclose(outfp);
    return 0;
}
```

What will the contents of `outfile.txt` be if the contents of `infile.txt` is `gdkkn`?

hello

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

9. Low-Level and Socket Programming. (15 marks)

(a) The C source file `sample.c` contains the following:

(2 marks)

```
#include <stdio.h>

int main(void)
{
#ifdef HELLO
    printf("hello");
#else
    printf("world");
#endif
    return 0;
}
```

If the source is compiled with the command

```
gcc sample.c -o sample
```

What is the output when `sample` is executed?

`world`

(b) Using a C structure, declare a bit-field consisting of the following fields:

(3 marks)

- version: 4 bits
- sequence: 2 bits

Use `magic_byte` as the structure tag.

```
struct magic_byte {
    char version : 4;
    char sequence : 2;
};
// data type can be unsigned char, int8_t, uint8_t
```

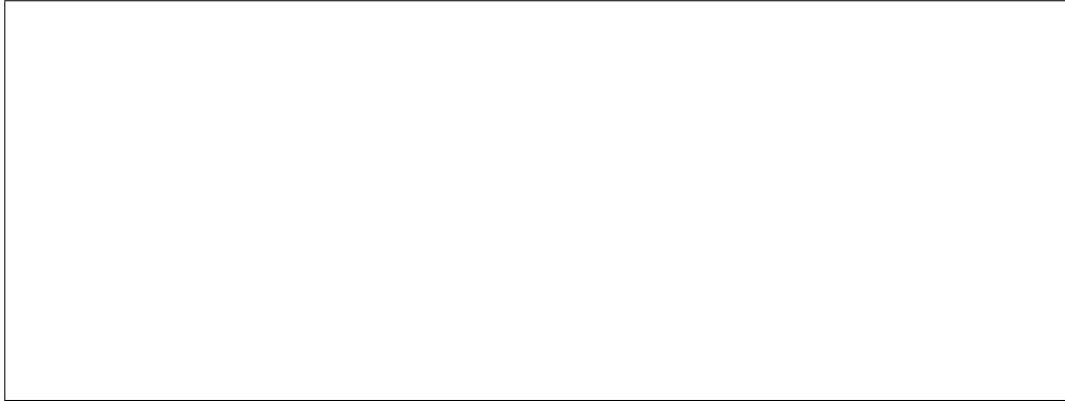
- (c) What are the two types of sockets supported by the `socket` system call?
(2 marks)

`SOCK_STREAM` (stream or TCP) and `SOCK_DGRAM` (data-gram or UDP)

- (d) Discuss briefly the steps involved in establishing a socket in a server process, stating the specific system call invoked in the step (if any). **(5 marks)**

1. Create a socket with the `socket()` system call.
2. Bind the socket to an address using the `bind()` system call.
3. Listen for connections with the `listen()` system call.
4. Accept a connection with the `accept()` system call.
5. Send and receive data.

- (e) Discuss briefly the steps involved in establishing a socket in a client process, stating the specific system call invoked in the step (if any). **(3 marks)**



1. Create a socket with the `socket()` system call.
2. Connect the socket to the address of the server using the `connect()` system call.
3. Send and receive data.

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

10. Process Management. (10 marks)

(a) Briefly explain the difference between a program and a process. **(2 marks)**

Program refers to executable code (typical stored on disk) while process is a program that is in execution.

(b) What are the four system calls for process management in C? **(4 marks)**

fork(), exec(), wait(), and exit()

(c) You are given the following C program:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <unistd.h>
4  #include <sys/wait.h>
5
6  int gvar = 2;
7
8  int main(void)
9  {
10     int lvar = 4;
11     pid_t pid;
12
13     if ((pid = fork()) < 0) {
14         printf("fork error\n");
15     }
16     if (pid == 0) {
17         gvar++;
18         lvar++;
19     } else {
20         wait(NULL);
```



```
21     }  
22  
23     printf("%ld %d %d\n", (long)getpid(), gvar, lvar);  
24     exit(0);  
25 }
```

i. Which line(s) are executed only in the child process? **(2 marks)**

Lines 17 and 18

ii. Assume that the fork is successful and that the parent process ID is 32346 while the child process ID is 32347. What is the output of the program? **(2 marks)**

```
32347 3 5  
32346 2 4
```

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID:

* * * * *