Family Name: ..... **First Name:** .....

Student ID: ..... **Signature** .....

NWEN241: Systems Programming

Mid-term Test

4 April 2025 ** WITH SOLUTIONS **

Instructions

- Time allowed: **120 minutes**
- There are **100** marks in total.
- Attempt **ALL** the multiple choice questions by writing the **letter** of the correct answer in the box provided.
- Only silent non-programmable calculators or silent programmable calculators with their memories cleared are permitted in this examination.
- Only paper (non-electronic) foreign to English language dictionaries are allowed.

Questions	Marks	
1. Introduction to Systems Programming	[10]	
2. C Fundamentals	[10]	
3. Function-Like Macros & One-Dimensional Arrays	[10]	
4. Multi-Dimensional Arrays & Strings	[10]	
5. Structures	[10]	
6. Pointers	[10]	
7. Storage Classes & Process Layout	[10]	
8. Dynamic Memory Management	[10]	
9. User Defined Types	[10]	
10. FILE Stream I/O	[10]	
	TOTAL:	

1. Introduction to Systems Programming

(i) **[2 marks]** Which of the following is NOT a systems program?

(a) Android operating system
(b) GDB debugger
(c) Web browser
(d) Device driver
(e) Virtual machine

c

(ii) **[2 marks]** Directive `#include <header.h>` makes the preprocessor search in which locations for `header.h` ?

(a) in pre-defined locations
(b) in current directory only
(c) in current directory, then in locations specified by the programmer
(d) in current directory, then in pre-defined locations
(e) in locations specified by the programmer, then in pre-defined locations

a

(iii) **[2 marks]** The compilation process of a C program involves the following sequence of phases:

(a) Compilation, Assembly, Linking, Preprocessing
(b) Preprocessing, Compilation, Assembly, Linking
(c) Assembly, Compilation, Preprocessing, Linking
(d) Preprocessing, Assembly, Linking, Compilation
(e) Linking, Preprocessing, Assembly, Compilation

b

(iv) **[2 marks]** A C header file should NOT contain:

(a) function prototype
(b) type definition
(c) constant definition
(d) variable declaration
(e) macro

d

(v) **[2 marks]** Which of the following is NOT a valid C identifier?

(a) invalid_order
(b) _coke_
(c) Frozen_L&P
(d) customer1
(e) boolean

c

2. C Fundamentals

(i) **[2 marks]** Which of the following C data types is NOT machine-dependent?

(a) `int`
(b) `short int`
(c) `char`
(d) `double`
(e) `float`

c

(ii) **[2 marks]** The literal 0x8feeU belongs to which of the following data type?

(a) `unsigned int`
(b) `int`
(c) `char`
(d) `double`
(e) `float`

a

(iii) **[3 marks]** A C program contains the following declarations:

```
int a, b;
long Ax ;
short sht;
float ff;
char cx;
```

What is the resulting data type of the following expression?

```
12 * a + (long) (Ax / sht) - ff * cx / b
```

(a) `int`
(b) `double`
(c) `long`
(d) `float`
(e) `short`

d

(iv) **[3 marks]** What would be the output from the given program?

```
int main() {
    int i=9;
    for(i--; i--; i--)
    printf("%d ", i);
return 0;
}
```

(a) 9 8 7 6 5 4 3 2 1
(b) 9 7 5 3 1
(c) 8 6 4 2
(d) 7 4 1
(e) 7 5 3 1

e

3. Function-Like Macros & One-Dimensional Arrays

(i) **[3 marks]** What is the output of the following program?

```
#include <stdio.h>
#define twice(x) x+x
#define thrice(y) (y+y)+(y)

int main()
{
  int x = 36/thrice(twice(6));
  printf("%d\n", x);
}
```

(a) 13
(b) 1
(c) 36
(d) 0
(e) 27

a

(ii) **[3 marks]** If we compile and execute the following program, what is the output?

```
#include <stdio.h>
#define MAX 1000
int main()
{
   int MAX = 100;
   printf("%d ", MAX);
}
```

(a) 1000
(b) 100
(c) Runtime Error
(d) Compilation Error
(e) Garbage output

d

(iii) **[2 marks]** The following macros increments the value by one.

```
#define INC1(a) ((a)+1)
#define INC2 (a) ((a)+1)
#define INC3( a ) (( a ) + 1)
#define INC4 ( a ) (( a ) + 1)
```

Which of the following statements is correct regarding the above macros?

(a) Only `INC1` is correct.
(b) All (i.e. `INC1`, `INC2`, `INC3` and `INC4`) are correct.
(c) Only `INC1` and `INC3` are correct.
(d) Only `INC1` and `INC2` are correct.
(e) None are correct.

c

(iv) **[2 marks]** Which of the following is a correct way to initialize an array in C?

(a) `int arr[] = {1, 2, 3, 4, 5;}`
(b) `int arr[4] = {0};`
(c) `int arr[3] = {0, 1, 2, 3};`
(d) (a), (b) and (c)
(e) (a) and (b)

b

4. Multi-Dimensional Arrays & Strings

(i) **[2 marks]** Given the following declarations:

```
char str1[]="cat";
char str2[]={'c','a','t'};
```

Which of the following is correct?

(a) Both `str1` and `str2` are character arrays.
(b) Both `str1` and `str2` are strings.
(c) `sizeof(str1) < sizeof(str2)`
(d) `sizeof(str1) == sizeof(str2)`
(e) `sizeof(str1) > sizeof(str2)`

e

(ii) **[2 marks]** When declaring a multi-dimensional array, which of the following statements is *true*?

(a) First dimension size is optional when initializing the array at the same time.
(b) Last dimension size is optional when initializing the array at the same time.
(c) All dimensions of a multidimensional array must be specified.
(d) Memory locations of elements of a multidimensional array are not sequential.
(e) None of the above.

a

(iii) **[3 marks]** Which of the following code snippets is *invalid* ?

(a) `char fish[20]; fish = "Kingfish";`
(b) `char fish[20]; strcpy(fish, "Kingfish");`
(c) `char fish[20] = "Kingfish";`
(d) `char fish[20]; char *kingy = "Kingfish"; int i; for (i=0; i<8; i++) fish[i]=*(kingy+i);`
(e) None of the above.

a

(iv) **[3 marks]** Given the following program:

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

int main()
{
    char p[20]; char *s = "nwen241";
    int length = strlen(s); int i;

    for (i = 0; i < length; i++)
        p[i] = s[length - i];
        printf("%s",p);
}
```

What does the program print?

(a) 142new
(b) 142newn
(c) No output is printed
(d) nwen241
(e) wen241

c

5. Structures

(i) **[2 marks]** Consider the following structure definition.

```
struct device {
    long IP_address;
    float utilisation;
};
```

What does the following C statement declare?

```
struct device *s[5];
```

(a) An array of size 5, each element is pointer to a structure of type `device`
(b) A structure of 2 fields, each field being a pointer to an array of 5 elements
(c) An array of size 5, each element of which is a structure of type `device`
(d) An array of size 5, each element is a structure of two pointers
(e) None of the above

a

(ii) **[2 marks]** What is actually passed if you pass a structure variable to a function?

(a) Copy of structure variable.
(b) Reference to a copy of the structure variable.
(c) Starting address of structure variable.
(d) Ending address of structure variable.
(e) All of the above.

a

(iii) **[2 marks]** Which of the following operations is illegal in structures?

(a) Pointer to a variable of the same structure.
(b) Dynamic allocation of memory for the structure.
(c) Static allocation of memory for the structure.
(d) Typecasting of structure.
(e) None of the above.

d

(iv) **[2 marks]** Which of the following statements about C structure elements is *correct*.

(a) Structure elements are stored on random free memory locations.
(b) Structure elements are stored in register memory locations.
(c) Structure elements are stored in contiguous memory locations.
(d) Size of the C structure is the size of the largest element.
(e) None of the above.

c

(v) **[2 marks]** What are the types of data allowed inside a structure?

(a) int, float, double, long double
(b) char, array, strings
(c) enum, union, same structure type members
(d) pointers
(e) All of the above.

e

6. Pointers

Consider the following 2D array declaration.

```
char m[4][6] = "ABCDE", "FGHIJ", "KLMNO", "PQRST";
```

(i) **[2 marks]** What is the value of `**m` ?

(a) A
(b) B
(c) C
(d) D
(e) E

a

(ii) **[2 marks]** What is the value of `*(*m+8)` ?

(a) F
(b) G
(c) H
(d) I
(e) J

c

(iii) **[2 marks]** What is the value of `*(m[1]+3)` ?

(a) D
(b) I
(c) S
(d) H
(e) \0

b

(iv) **[2 marks]** What is the value of `(*(m+2))[2]` ?

(a) T
(b) N
(c) \0
(d) G
(e) M

e

(v) **[2 marks]** What is the value of `*(&m[0][0]+21)` ?

(a) R
(b) T
(c) P
(d) S
(e) \0

d

7. Storage Classes & Process Layout

(i) **[2 marks]** In C, generic pointers can be declared with

(a) static
(b) void
(c) extern
(d) const
(e) None of the above.

b

(ii) **[2 marks]** In the following declaration:

```
register int i;
```

Which of the following is *true* ?

(a) The value of variable `i` is guaranteed to be stored in a CPU register.
(b) If registers are all allocated, the compiler will store `i` in cache memory.
(c) A register variable is local to the block which contains it.
(d) The compiler can ignore the request, in which case the storage class defaults to `static`.
(e) None of the above.

c

(iii) **[2 marks]** Which of the following Storage Class declarations is *optional* ?

(a) `auto`
(b) `extern`
(c) `register`
(d) `static`
(e) `void`

a

(iv) **[2 marks]** Which C Storage Class variables are stored in the `DATA` segment?

(a) `extern` only
(b) `static` only
(c) `extern` and `register`
(d) `static` and `register`
(e) `extern` and `static`

e

(v) **[2 marks]** Consider the following C program.

```
int func(int d) {
    int b;
    {
        int c;
    }
}
int a;
int main() {
    func(a);
}
```

What will be the sequence of allocation and deletion of variables in the above code?

(a) Allocate `a`, `b`, `c`, `d` ; Deallocate `a`, `b`, `c`, `d`;
(b) Allocate `a`, `b`, `c`, `d` ; Deallocate `d`, `c`, `b`, `a`;
(c) Allocate `a`, `d`, `b`, `c` ; Deallocate `c`, `b`, `d`, `a`;
(d) Allocate `a`, `d`, `b`, `c` ; Deallocate `c`, `d`, `b`, `a`;
(e) All the above are incorrect.

c

8. Dynamic Memory Management

(i) **[2 marks]** With every use of a memory allocation function, what function should be used to release allocated memory which is no longer needed?

(a) `dealloc()`
(b) `release()`
(c) `free()`
(d) `unalloc()`
(e) `realloc()`

c

(ii) **[2 marks]** Consider the following code snippet. Assuming the allocation is successful, the size (in bytes) of the memory block pointed to by `cp` will be:

```
char *cp;
cp = (char *)malloc(20*sizeof(char));
```

(a) 4 bytes
(b) 20 bytes
(c) 40 bytes
(d) 80 bytes
(e) `sizeof(*cp)`

b

(iii) **[2 marks]** Consider the following code snippet.

```
char *ptr = (char *)malloc(8*sizeof(char));
realloc(ptr, 12*sizeof(char));
```

After a successful call to `realloc()` on the second line, where does `ptr` point to?

(a) To previously allocated memory that has been released.
(b) To previously allocated memory that has been expanded to 12 bytes.
(c) NULL.
(d) To the newly allocated memory for 12 bytes in a different location.
(e) To previously allocated memory that the program can still use.

a

(iv) **[2 marks]** Which of the following is equivalent to the call `malloc(10*sizeof(double))` ?

(a) `calloc(10*sizeof(double))`
(b) `calloc(10)`
(c) `calloc(sizeof(double))`
(d) `calloc(10, sizeof(double))`
(e) None of the above.

d

(v) **[2 marks]** What is **NOT** a good reason for using `calloc()` to dynamically allocate memory for an array?

(a) When the array is large and may exceed the size of the Stack memory.
(b) To automatically deallocate the array when the function returns.
(c) To provide more flexibility to grow or shrink the size of the array.
(d) The size of the array is not known until runtime.
(e) The elements of the array are of unequal size.

b

9. User-defined Types

(i) **[4 marks]** What is the output of the following C program?

```
#include<stdio.h>
enum random { a, b = 9, c, d = -1, e};
int main()
{
    printf("%d %d %d %d %d\n",a,b,c,d,e);
}
```

(a) 0 9 10 -1 0
(b) 0 9 10 -1 1
(c) 0 9 10 -1 -2
(d) 0 9 1 -1 0
(e) 0 9 1 -1 -2

a

(ii) **[2 marks]** Which of the following declarations is *invalid* ?

(a) `typedef struct {float alcohol_content; long shelf_life;} wine;`
(b) `enum {red, white, rosé, sparkling};`
(c) `struct wine_stock {int alcohol_content; long shelf_life;}`
(d) `union {char wine_name[10]; long shelf_life;};`
(e) `enum white_wine {Riesling, Chardonnay, PinotGris, Muscat};`

d

(iii) **[2 marks]** Consider the following code snippet:

```
union {
    char C;
    short A;
    long L;
    float F;
} M;
M.A = 255;
```

The size of the variable `M` is equal to which of the following?

(a) `sizeof(char)`
(b) `sizeof(short)`
(c) `sizeof(float)`
(d) maximum (`sizeof(C),sizeof(A),sizeof(L),sizeof(F)`)
(e) None of the above.

d

(iv) **[2 marks]** Consider the following declaration:

```
union U {
    char C;
    short A;
    long L;
    float F;
};
```

What is the size of the memory allocated by the C compiler?

(a) `sizeof(char)`
(b) `sizeof(short)`
(c) `sizeof(float)`
(d) maximum (`sizeof(C),sizeof(A),sizeof(L),sizeof(F)`)
(e) None of the above.

e

10. FILE Stream I/O

(i) **[2 marks]** In C, which of the following will read a character from keyboard and store it in a character variable `c`?

(a) `gets(c);`
(b) `c = getc();`
(c) `gets(&c);`
(d) `getchar(&c);`
(e) `c = getchar();`

e

(ii) **[2 marks]** Consider the following C code snippet.

```
char c;
FILE *infp = fopen("infile.txt", "r");
FILE *outfp = fopen("outfile.txt", "w");
while( (c=getc(infp)) != EOF ) { putc(--c, outfp); }
fclose(infp); fclose(outfp);
```

If the contents of `infile.txt` is `Dpnqvufs`
What would be the contents of `outfile.txt` ?

(a) `Amknsrcp`
(b) `Dpnqvufs`
(c) `Computer`
(d) `@ljmrqbo`
(e) Empty file.

c

(iii) **[3 marks]** Consider the following C code snippet:

```
int i;
FILE *fp = fopen("input.txt", 'r');
fscanf(fp, "%d", &i);
/* Done reading from the file */
printf("%d", i);
```

Which of the following describes an issue (error and poor programming) with the code?

(a) The mode argument of `fopen()` should be a string, and `'r'` should be replaced by `"r"`.
(b) After the call to `fopen()`, its return value must be checked to ensure that the file opening was successful.
(c) After the call to `fscanf()`, its return value must be checked to ensure that the reading was successful.
(d) The file must be closed using `fclose()`.
(e) All of the above.

e

(iv) **[3 marks]** Which of the following statements is *false* ?

(a) To be able to read keyboard input, a program must first open the `stdin` stream.
(b) The function `fflush()` only works on streams that are open for output.
(c) The function `fscanf()` will return `EOF` if the end of file is reached, or errors were encountered while reading the file.
(d) When a binary file is opened with mode `"rb"`, the file must exist, otherwise, `fopen()` will return `NULL`.
(e) The call `rewind(fp)` is equivalent to the call `fseek(fp, -s, SEEK_END)`, where `s` is the size of the file (in bytes).

a

* * * * * * * * * * * * * * *

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.