Question 1

What is meaning of following declaration?

*int(*p[5])();*

- p is pointer to function
- p is array of pointer to function
- p is pointer to such function which return type is array
- p is pointer to array of function
- none of the above

Question 2

1. Which best describes the type of `x` in the following C declaration?

```
int (*x[10])(char *);
```

(a) a pointer to a function that takes an array of 10 strings and returns an `int`
(b) an array of 10 pointers to functions that take strings and return `int`s
(c) a pointer to a function that takes a string and returns an array of 10 `int`s
(d) a function that takes an array of 10 strings and returns an `int`

https://moss.cs.iit.edu/cs351/exams/midterm2019fall.pdf

Question 3

int *a[]

int (*a)[]

int* (*a)()

int* ((a())[])()

int (*(*a())[])()

int* (*(*a[])())[]

Solutions

int *a[] : array[] of pointer to int

int (*a)[] : pointer to array[] of int

int* (*a)() : pointer to function which returns pointer to int

int* ((a())[])() : function which returns array[] of functions that return pointer to int

int (*(*a())[])() : function which returns pointer to array of pointers to functions which return pointer to int

int* (*(*a[])())[] : array of pointer to function which returns pointer to array of pointer to int

Question 4

Which best describes the type of x in the following C declaration?

```
char *(*(*x)[10])()
```

(a) a pointer to an array of 10 pointers to functions returning strings
(b) a pointer to a function that takes an array of 10 void pointers and returns
(c) an array of 10 pointers to functions returning strings
(d) an array of functions that take arrays of 10 character pointers a string

http://www.cs.iit.edu/~nsultana1/teaching/F22CS351/past_exams/midterm-2020spring.pdf

Question 5

Which of the following function declaration is correct

```
#include<stdio.h>

int main()
{
    int arr[5][6];
    avg(arr);
    return 0;
}
```

a) `void avg(int *a[5][6])`
`{`
`}`
	b) `void avg(int a[][6])`
	`{`
	`}`

c) `void avg(int *a[][6])`
`{`
`}`
	d) `void fun(int a[][4])`
	`{`
	`}`

https://s3.amazonaws.com/serverless-libreoffice-pdff/563PROGRAMFILE1.pdf

Question 6

```
int main() {
    char  a[3] = {'p', 'q', 'r'};
    char  b[3] = {'s', 't', 'u'};
    char  c[3] = {'w', 'x', 'z'};
    char *d[3] = {a, b, c};
    char *e = d[1];
    char *f = e + 2;
    char g = *f;
    printf("%c\n", g);
    return 0;
}
```

2. Consider the following program.

```
int main() {
    char  a[3] = {'p', 'q', 'r'};
    char  b[3] = {'s', 't', 'u'};
    char  c[3] = {'w', 'x', 'z'};
    char *d[3] = {a, b, c};
    char *e = d[1];
    char *f = e + 2;
    char g = *f;
    printf("%c\n", g);
    return 0;
}
```

Note: `printf("%c\n", g);` means "print the character stored in variable `g`", so if `g='a'` it would print `a`.

What get prints out? If the program would crash write `crash`.

Solution: key: u

Question 7

```
int main() {
    int x[6] = {11, 12, 13, 14, 15, 16};
    int y[2] = {21, 22};
    int *z[2] = {x, y};
    int *w = z[0] + 3;
    int a = *w;
    printf("%d", a);
    return 0;
}
```

https://www.cs.virginia.edu/~jh2jf/courses/cs2130/spring2023/exams/f2022e2key.pdf

6. [8 points] Consider the following main function:

```
int main() {
    int x[6] = {11, 12, 13, 14, 15, 16};
    int y[2] = {21, 22};
    int *z[2] = {x, y};
    int *w = z[0] + 3;
    int a = *w;
    printf("%d", a);
    return 0;
}
```

What is printed? If the program would crash or seg fault, write **crash**.

Answer

14

Question 8

```c
#include<stdio.h>
void swap(int **x, int i, int j){
    int *temp;
    temp = x[i];
    x[i] = x[j];
    x[j] = temp;
}
int main()
{
    int a[6] = {1, 2, 3};
    int (*p)[6] = &a;
    int *arr[3] = {a,a+2,a+4};
    p[0][2] = 5;
    swap(arr,1,2);
    printf("%d\n", *arr[2]);
    return 0;
}
```

Pointer Arithmetic: Combining * and ++/--

- *p++; value: *p, inc: p
- (*p)++; value: *p, inc: *p
- ++(*p); value: (*p)+1, inc: *p
- *++p; value: *(p+1), inc: p

*p++ vs (*p)++ ?

- x = *p++ ⇒ x = *p ; p = p + 1;
- x = (*p)++ ⇒ x = *p ; *p = *p + 1;

https://cs.brynmawr.edu/Courses/cs246/spring2013/slides/07PassingPointers.pdf

Operator Precedents

- Unary operators associate right to left

```
y = *&x;          /* same as y = *(&x) */
```

- Unary operators bind more tightly than binary ones

```
y = *p + 1;       /* same as y = (*p) + 1; */
```

- More examples

```
y = *p++;         /* same as y = *p; p++; */
y = *(p++);       /* same as above */
y = *++p;         /* same as p++; y = *p; */
y = ++*p;         /* same as y = (*p) + 1; */
```

- When in doubt, liberally use parentheses

17

https://www.cs.princeton.edu/courses/archive/fall04/cos217/lectures/05pointers.pdf

A portion of the C Operator Precedence Table

Operator	Associativity
++ postfix increment	L to R
-- postfix decrement	
[] array element	
() function call	
* indirection	R to L
++ prefix increment	
-- prefix decrement	
& address-of	
sizeof size of type/object	
(type) type cast	
* multiplication	L to R
/ division	
% modulus	
+ addition	L to R
- subtraction	
.	
.	
= assignment	R to L

https://cseweb.ucsd.edu/~ricko/CSE30/Midterm.fa12.pdf

Operator Precedence table (only top 2 rows)

	Operators	Associativity
1	`( )` `[ ]` `->` `.` `++` `--` (postfix)	left to right
2	`sizeof` `&` `*` `+` `-` `~` `!` *typecasts* `++` `--` (prefix)	right to left

Question 9

```c
#include <stdio.h>


int main() {
    int A[] = {4, 0, 5, 9, 2};
    int *ptr = &A[2];
    printf("%d", *ptr++);

    return 0;
}
```

A. 9
B. 5
C. 6
D. 0

Solution

A. is wrong because it incremented ptr to point to 9 and dereferenced the value. This would be correct is the statement was printf("%d", *++ptr).

B. is correct ptr is post-incremented, so it would return its de-referenced value first in the printf() statement before incrementing to point to 9.

C. is wrong because it incremented the dereferenced value of ptr. This would be correct is the statement was printf("%d", ++*ptr).

D. is wrong. May have mistaken the index of the beginning of the array as 1 instead of 0. It would be correct if int *ptr = &A[1] using the same logic as A.

Question 10

```
int n = 3;
int *p = &n;
printf("n=%d\n", ++*p);
printf("n=%d\n", n++);
```

https://ee209-2020-fall.github.io/oldmidterm/fall11exam_KAIST.pdf

Question 11

```c
#include <stdio.h>
int main()
{
    char data = 'a';
    char *ptr = &data;
    printf("%c", ++*ptr++);

    return 0;
}
```

http://www.cs.columbia.edu/~janak/teaching/s04-cs10034/lectures/CS1113-Outline.pdf

Question 12

If I have the following C program

```
int main(void) {
      int a = 15;
      int b = 12;
      int *p = &a;
      int *q = p;
      char *cp = (char*) &a;
      ...
      *p++;
      q++;
      cp += 2;
      ...
}
```

what value would you see in each case if you printed each of the following?

*p _____ *q _____

a _____ b _____

*cp _____

Question 13

What would be the output of the following code?

```
void main()
{
char *p = "UNIVERSITY";
printf("%c",++*(p++));
}
```

a) No Output b)Error
c) N d) V

Question 14

What is the output of the following code?

```
#include<stdio.h>
void abc(char[]);
int main()
{
        char arr[100];
        arr[0] = 'a';
        arr[1] = 'b';
        arr[2] = 'c';
        arr[4] = 'd';
        abc(arr);
        return 0;
}
void abc(char arr[])
{
        printf("%c", *++arr);
        printf("%c", *arr++);
}
```

A. ab
B. bb
C. bc
D. cc

Question 15

Q: Does the following code give an error at run time? If yes, why? If no, what is the output?

```
main() {
    int *p;
    *p = 5;
    printf("%d",(*p)++);
}
```

A. Yes, because the pointer does not point to a valid address in memory

B. Yes, because there is a syntax error in the printf statement: `(*p)++` is not a valid operation on a pointer

C. No, the output of the program is the value pointed to by p, which is 5

D. No, the output of the program is the value pointed to by p, plus one, which is 6

https://cseweb.ucsd.edu/classes/wi14/cse30-c/lectures/PI_WI_14_CSE30_lecture_4_post.pdf

Question 16

```c
int main() {
    char *p, *q, y = 0;
    char x[8] = {0, 1, 2, 3, 4, 5, 6, 7};
    int i = 0;

    p = x;
    q = &x[6];
    *q = 'a';

    for (i = 0; i < 6; i++) {
        *p++ = *q;
    }

    q = p;

    printf("%d", x[1]);
    return 0;
}
```

Question 17

```
char *p = "1234";
while(*p)
   printf("%c ", *++p);
```

Question 18

```
char *p = "1234";
while(*p)
   printf("%c ", *p++);
```

Question 19

Given the declaration:

```
int num = 6;
int *p = &num;
```

which statement below increments the value of num?

- ○ *p++;
- ○ (*p)++;
- ○ (*num)++;
- ○ p++;

Question 20

```
#include <stdio.h>

int main ()
{
    int i, *p, count = 0;
    p = &count;

    for (i = 0; i < 7; i++) {
        count++;
        (*p)++;
    }

    printf("count = %d, Have a great day.\n", count);
    return 0;
}
```

What is the output of the program?

Question 21

What is the output of the following program:

```
#include <stdio.h>

int *confuse(int *x, int *y) {
    (*y)++;
    y = x;
    *y = 10;
    return (y);
}

int main(void) {
    int a = 6, b = 7;
    int *f = &b;

    f = confuse(&a, &b);
    (*f)++;

    printf("a = %d and b = %d\n", a, b);
    return 0;
}
```

http://exams.skule.ca/exams/APS105H1_20181_6215560621042018-midterm-solution.pdf

Question 22

```c
#include <stdio.h>

int main()
{
    int *pv, v[] = {1, 4, 7, 10, 13, 16, 19};
    char *pc, c[] = "FEDCBA";

    pv = v + 4;
    printf("%d \n ", (*pv)++);
    printf("%d \n", ++*pv);

    pv = v;
    printf("%d \n ", *++pv);
    printf("%d \n", pv[1]);

    return 0;
}
```

https://staff.emu.edu.tr/omarramadan/en/Documents/cmpe112sp2017%20final%20with%20solutions.pdf

Question 23

What's the output of this code snippet?

```
int a[10], i, *p =a;

for (i =0; i < 10; i++)
      a[i] = i;

printf("value 1 = %d\n", *p++);
printf("value 2 = %d\n", (*p)++);
printf("value 3 = %d\n", (*(p+4))--);
printf("value 4 = %d\n", *--p);
printf("value 5 = %d\n", ++*p);
```

Output:

value 1 = _____

value 2 = _____

value 3 = _____

value 4 = _____

value 5 = _____

https://ee209-2020-fall.github.io/oldmidterm/fall17exam_KAISTans.pdf

Question 24

```
int x[] = { 2, 4, 6, 8, 10 };
int *p = x;
int **pp = &p;
(*pp)++;
(*(*pp))++;
printf("%d\n", *p);
```

Result is:
A: 2
B: 3
C: 4
D: 5
E: None of the above

https://inst.eecs.berkeley.edu//~cs61c/sp20/pdfs/lectures/lec03.pdf

Question 25

```c
#include <stdio.h>

int a = 380, b = 480, c = 42;

void f(int *x, int *y, int z) {
  ++(*x);
  ++(*y);
  ++z;
}


int main() {
  printf("a=%d b=%d c=%d\n", a, b, c);
  f(&a, &b, c);
  printf("a=%d b=%d c=%d\n", a, b, c);

  return 0;
}
```

Question 26

The output of the following code is 5 5.
Which of the following correctly replaces the missing lines in the code?

```
int main() {
    int a, *ptr, arr[12] = {1, 2,1,2};
    ptr = arr;
    //line 1
    printf("%d", a);
    //line 2
    printf("%d", a);
    return 0;
}
```

A. Line 1: a = *ptr++ + 3; and Line 2: a = *ptr++ + 3;

B. Line 1: a = *ptr++ + 3; and Line 2: a = ++*ptr + 3;

C. Line 1: a = ++*ptr + 3; and Line 2: a = ++*ptr + 3;

D. Line 1: a = ++*ptr + 3; and Line 2: a = *ptr++ + 3;

Question 27

Consider the code.
However, part of the code is missing.

The code is supposed to give the output -
7 26 8 6 17 4 32 1

What can the missing parts be?

```
#include <stdio.h>
int main() {
    int k, *p;
    int v[2][4] = {7, 3, 8, 6, 9, 4, 5, 1};

    v[1][0] =           ;       // Missing_1
    v[0][1] =           ;       // Missing_2
    *(v[1] + 2) =       ;       // Missing_3

    p = (int *)v;
    for (k = 0; k < 8; k++)
        printf("%d ", p[k]);

    return 0;
}
```

A. Missing_1: 26 Missing_2: 17 Missing_3: 32
B. Missing_1: 17 Missing_2: 32 Missing_3: 26
C. Missing_1: 17 Missing_2: 26 Missing_3: 32
D. Missing_1: 32 Missing_2: 26 Missing_3: 17

Question 28

```c
#include <stdio.h>

void fun(char *p, char *r)
{
    while(*p!='\0'){
        *p++ = *r++;
    }
}


int main() {

    char x[]="hello cat";
    char y[]="guten tag";

    fun(x,y);
    printf("%s", x);

    return 0;
}
```

Question 29

```c
int main()
{
    char a[] = "CSE30";
    char *p = a;

    printf("%c\n", *p++);
    *(p+2) = p[3];

    printf("%c\n", *(p+2));
    *(p+3) = p[-1];

    printf("%c\n", *(p+3));
    printf("%c\n", *++p);

    return 0;
}
```

https://cseweb.ucsd.edu/~ricko/CSE30/Midterm.fa12.pdf

Question 30

What is the output of the following program?

```
int f(int x, int *y) {
   x += 2;  *y += 1;
   return x + *y;
    }
int g(int *x, int y) {
   y = ++*x;
   return *x + y;
    }
int main( ) {
   int x = 2, y = 3;
   printf("%d   ",f(x, &y));
   printf("%d   ",g(&x, y));
   printf("%d   %d \n",x, y);
      return 0;
}
```

(A) 8 6 3 4

(B) 8 10 4 4

(C) 8 6 3 3

(D) 8 10 3 4

Question 31

Let f be the following C function:

```
void f(char *p)
{
  char *q = p;

  while (*q)
    q++;
  while (p < q) {
    char ch = *p;
    *p++ = *--q;
    *q = ch;
  }
}
```

What modification does f perform to the string that is passed to it?

Question 32

What will be the contents of the **a** array after the following statements are executed?

```
#define N 10

int main() {
    int a[N] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    int *p = &a[0];
    int *q = &a[N - 1];
    int temp;

    while (p < q) {
        temp = *p;
        *p++ = *q;
        *q-- = temp;
    }

    return 0;
}
```

- Lesson?
  - Using anything but the standard `*p++`, `(*p)++` causes more problems than it solves!

https://inst.eecs.berkeley.edu/~cs61c/fa13/lec/04LecF13CIntroIIx2.pdf