Pointers

Total points 69/69

www.learn-in-depth.com

Email *

abdallah.shabaan.ghazy@gmail.com

✓ Q45) * 1/1

In the following program add a statement in the function `fact()` such that the factorial gets stored in `j`.

```
#include<stdio.h>
void fact(int*);

int main()
{
    int i=5;
    fact(&i);
    printf("%d\n", i);
    return 0;
}
void fact(int *j)
{
    static int s=1;
    if(*j!=0)
    {
        s = s**j;
        *j = *j-1;
        fact(j);
        /* Add a statement here */
    }
}
```

- ○ A. j=s;
- ◉ B. *j=s; ✓
- ○ C. *j=&s;
- ○ D. &j=s;

✓ Q51) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
   int a = 320;
   char *ptr;
   ptr =( char *)&a;
   printf("%d ",*ptr);
   return 0;
```

- (A) 2
- (B) 320
- (C) 64 ✓
- (D) Compilation error
- (E) None of above

✓ Q34)What will be the output of the program assuming that the array begins at the location 1002 and size of an integer is 4 bytes? *1/1

```
#include<stdio.h>

int main()
{
    int a[3][4] = { 1, 11, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };
    printf("%u, %u, %u\n", a[0]+1, *(a[0]+1), *(*(a+0)+1));
    return 0;
}
```

- A. 448, 4, 4
- B. 520, 2, 2
- C. 1006, 2, 2
- D. Error
- E. 1006,11,11 ✓

✓ Q30) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    int i=3, *j, k;
    j = &i;
    printf("%d\n", i**j*i+*j);
    return 0;
}
```

- (●) A. 30 ✓
- ( ) B. 27
- ( ) C. 9
- ( ) D. 3

✓ Q7)Assume that an int variable takes 4 bytes and a char variable takes 1 byte *1/1

```
#include<stdio.h>
int main()
{
    int arr[] = {10, 20, 30, 40, 50, 60};
    int *ptr1 = arr;
    int *ptr2 = arr + 5;
    printf("Number of elements between two pointer are: %d.",
                                (ptr2 - ptr1));
    printf("Number of bytes between two pointers are: %d",
                               (char*)ptr2 - (char*) ptr1);
    return 0;
}
```

- (A) Number of elements between two pointer are: 5.Number of bytes between two pointers are: 20 ✓
- (B) Number of elements between two pointer are: 20.Number of bytes between two pointers are: 20
- (C) Number of elements between two pointer are: 5.Number of bytes between two pointers are: 5
- (D) Compiler Error
- (E) Runtime Error

✓ Q52) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
    int i = 3;
    int *j;
    int **k;
    j=&i;
    k=&j;
    printf("%u %u %d ",k,*k,**k);
    return 0;
}
```

- (A) Address, Address, 3 ✓
- (B) Address, 3, 3
- (C) 3, 3, 3
- (D) Compilation error
- (E) None of above

✓ Q70) * 1/1

```c
#include<stdio.h>
int check (int, int);

int main()
{
    int c;
    c = check(10, 20);
    printf("c=%d\n", c);
    return 0;
}
int check(int i, int j)
{
    int *p, *q;
    p=&i;
    q=&j;
    i>=45 ? return(*p): return(*q);
}
```

- A. Print 10
- B. Print 20
- C. Print 1
- D. Compile error ✓

✓ Q33) * 1/1

What will be the output of the program ?

```
#include<stdio.h>
int *check(static int, static int);

int main()
{
    int *c;
    c = check(10, 20);
    printf("%d\n", c);
    return 0;
}
int *check(static int i, static int j)
{
    int *p, *q;
    p = &i;
    q = &j;
    if(i >= 45)
        return (p);
    else
        return (q);
}
```

- A. 10
- B. 20
- C. Error: Non portable pointer conversion
- D. Error: cannot use static for function parameters ✓

✓ Q43) * 1/1

In the following program add a statement in the function `fun()` such that address of `a` gets stored in `j`?

```
#include<stdio.h>
int main()
{
    int *j;
    void fun(int**);
    fun(&j);
    return 0;
}
void fun(int **k)
{
    int a=10;
    /* Add a statement here */
}
```

- A. **k=a;
- B. k=&a;
- C. *k=&a ✓
- D. &k=*a

✓ Q48) * 1/1

Is there any difference between the following two statements?

```
char *p=0;
char *t=NULL;
```

A. Yes

B. No

- A
- B ✓

✓ Q54) * 1/1

What will be output of following program?

```
#include<stdio.h>
#include<string.h>
int main(){
     register a = 25;
     int far *p;
     p=&a;
     printf("%d ",*p);
     return 0;
}
```

- (A) 25
- (B) 4
- (C) Address
- (D) Compilation error ✓
- (E) None of above

✓ Q6) * 1/1

Assume that float takes 4 bytes, predict the output of following program.

```
#include <stdio.h>

int main()
{
    float arr[5] = {12.5, 10.0, 13.5, 90.5, 0.5};
    float *ptr1 = &arr[0];
    float *ptr2 = ptr1 + 3;

    printf("%f ", *ptr2);
    printf("%d", ptr2 - ptr1);

   return 0;
}
```

- (●) 90.500000 3 ✓
- ( ) 90.500000 12
- ( ) 10.000000 12
- ( ) 0.500000 3

✓ Q15) * 1/1

```
#include<stdio.h>

void swap (char *x, char *y)
{
    char *t = x;
    x = y;
    y = t;
}

int main()
{
    char *x = "geeksquiz";
    char *y = "geeksforgeeks";
    char *t;
    swap(x, y);
    printf("(%s, %s)", x, y);
    t = x;
    x = y;
    y = t;
    printf("n(%s, %s)", x, y);
    return 0;
}
```

(geeksquiz, geeksforgeeks)
(geeksforgeeks, geeksquiz)

◉ A ✓

(geeksforgeeks, geeksquiz)
(geeksquiz, geeksforgeeks)

○ B

C

```
(geeksquiz, geeksforgeeks)
(geeksquiz, geeksforgeeks)
```

D

```
(geeksforgeeks, geeksquiz)
(geeksforgeeks, geeksquiz)
```

✓ Q26 * 1/1

Consider the following variable declarations and definitions in C

```
i) int var_9 = 1;
ii) int 9_var = 2;
iii) int _ = 3;
```

- (●) Both i) and iii) are valid. ✓
- ( ) Only i) is valid.
- ( ) Both i) and ii) are valid.
- ( ) All are valid.

✓ Q58) * 1/1

What will be output of following program?

```
#include<stdio.h>
#include<string.h>
int main(){
     int a = 5,b = 10,c;
     int *p = &a,*q = &b;
     c = p - q;
     printf("%d" , c);
     return 0;
}
```

- (A) 1 ✓
- (B) 5
- (C) -5
- (D) Compilation error
- (E) None of above

✓ Q66) * 1/1

What will be output when you will execute following c code?

```
#include<stdio.h>
void main(){
    static int a=2,b=4,c=8;
    static int *arr1[2]={&a,&b};
    static int *arr2[2]={&b,&c};
    int* (*arr[2])[2]={&arr1,&arr2};
    printf("%d %d\t",*(*arr[0])[1],  *(*(**(arr+1)+1)));
}
```

- (A) 2 4
- (B) 2 8
- (C) 4 2
- (D) 4 8 ✓
- (E) None of the above

✓ Q38) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    char str[] = "peace";
    char *s = str;
    printf("%s\n", s++ +3);
    return 0;
}
```

- [A]. peace
- [B]. eace
- [C]. ace
- [D]. ce ✓
- [E]. e

✓ Q61) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
     int i = 5;
     int *p;
     p = &i;
     printf(" %u %u", *&p , &*p);
     return 0;
}
```

- A) 5 Address
- (B) Address Address ✓
- (C) Address 5
- (D) Compilation error
- (E) None of above

✓ Q20) * 1/1

What will be the output produced by the following C code:

```
int main()
{
    int array[5][5];
    printf("%d",( (array == *array) && (*array == array[0]) ));
    return 0;
}
```

- (●) 1 ✓
- ( ) 0
- ( ) 2
- ( ) -1

✓ Q44) * 1/1

Which of the statements is correct about the program?

```c
#include<stdio.h>

int main()
{
    int arr[3][3] = {1, 2, 3, 4};
    printf("%d\n", *(*(*(arr))));
    return 0;
}
```

- ○ [A]. Output: Garbage value
- ○ [B]. Output: 1
- ○ [C]. Output: 3
- ◉ [D]. Error: Invalid indirection ✓

✓ Q29) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    static char *s[] = {"black", "white", "pink", "violet"};
    char **ptr[] = {s+3, s+2, s+1, s}, ***p;
    p = ptr;
    ++p;
    printf("%s", **p+1);
    return 0;
}
```

- (●) [A] ink ✓
- ( ) [B]. ack
- ( ) [C]. ite
- ( ) [D]. let

✓ Q24)What does the following expression means ? char ∗(∗(∗ a[N]) ( )) ( ); * 1/1

- ( ) a pointer to a function returning array of n pointers to function returning character pointers.
- ( ) a function return array of N pointers to functions returning pointers to characters
- ( ) an array of n pointers to function returning pointers to characters
- (●) an array of n pointers to function returning pointers to functions returning pointers to characters. ✓
- ( ) all of them

✓ Q25) * 1/1

```c
#include "stdio.h"
int main()
{
 void *pVoid;
 pVoid = (void*)0;
 printf("%lu",sizeof(pVoid));
 return 0;
}
```

- ○ Assigning (void *)0 to pVoid isn't correct because memory hasn't been allocated. That's why no compile error but it'll result in run time error.
- ○ Assigning (void *)0 to pVoid isn't correct because a hard coded value (here zero i.e. 0) can't assigned to any pointer. That's why it'll result in compile error.
- ◉ No compile issue and no run time issue. And the size of the void pointer i.e. pVoid would equal to size of int. ✓
- ○ sizeof() operator isn't defined for a pointer of void type.

✓ Q56) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
    char arr[10];
    arr = "world";
    printf("%s",arr);
    return 0;
}
```

- (A) world
- (B) w
- (C) Null
- (D) Compilation error ✓
- (E) None of above

✓ Q59) * 1/1

What will be output of following program?

```
#include<stdio.h>
unsigned long int (* avg())[3]{
    static unsigned long int arr[3] = {1,2,3};
    return &arr;
}
int main(){
    unsigned long int (*ptr)[3];
    ptr = avg();
    printf("%d" , *(*ptr+2));
    return 0;
}
```

- (A) 1
- (B) 2
- (C) 3 ✓
- (D) Compilation error
- (E) None of above

✓ Q62) * 1/1

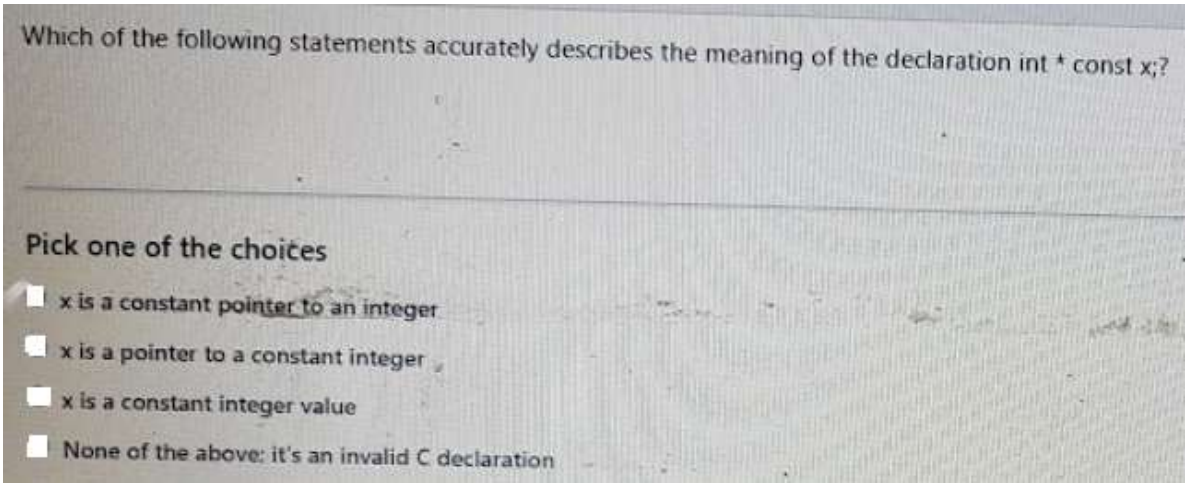
```
(q) What will be output of following c code?
void main()
{
     struct field
     {
          int a;
          char b;
     }bit;
     struct field bit1={5,'A'};
     char *p=&bit1;
     *p=45;
     clrscr();
     printf("\n%d",bit1.a);
     getch();
}
```

- 5
- 45 ✓
- 0
- None of above

✓ Q23) The following statement in 'C' int (*f())[ ]; declares * 1/1

- a function returning a pointer to an array of integers. ✓
- a function returning an array of pointers to integers.
- array of functions returning pointers to integers.
- an illegal statement.

✓ Q1) * 1/1


Which of the following statements accurately describes the meaning of the declaration int * const x;?
Pick one of the choices
x is a constant pointer to an integer
x is a pointer to a constant integer
x is a constant integer value
None of the above: it's an invalid C declaration

- (●) A ✓
- ( ) B
- ( ) C
- ( ) D

✓ Q9) * 1/1

```
int main()
{
 char *ptr = "GeeksQuiz";
 printf("%cn", *&*&*ptr);
 return 0;
}
```

- ○ Compiler Error
- ○ Garbage Value
- ○ Runtime Error
- ◉ G ✓
- ○ GeeksQuiz

✓ Q65) * 1/1

What will be output when you will execute following c code?

```
#include<stdio.h>
void main(){
    int array[2][3]={5,10,15,20,25,30};
    int (*ptr)[2][3]=&array;
    printf("%d\t",***ptr);
    printf("%d\t",***(ptr+1));
    printf("%d\t",**(*ptr+1));
    printf("%d\t",*(*(*ptr+1)+2));
}
```

- (A) 5 Garbage 20 30 ✓
- (B) 10 15 30 20
- (C) 5 15 20 30
- (D) Compilation error
- (E) None of the above

✓ Q53) * 1/1

What will be output of following program?

```
#include<stdio.h>
#include<string.h>
int main(){
     char *ptr1 = NULL;
     char *ptr2 = 0;
     strcpy(ptr1," c");
     strcpy(ptr2,"questions");
     printf("\n%s %s",ptr1,ptr2);
     return 0;
}
```

- (A) c questions
- (B) c (null)
- (C) (null) (null) ✓
- (D) Compilation error
- (E) None of above

✓ Q41) * 1/1

Point out the compile time error in the program given below.

```
#include<stdio.h>

int main()
{
    int *x;
    *x=100;
    return 0;
}
```

- A. Error: invalid assignment for x
- B. Error: suspicious pointer conversion
- C. No error ✓
- D. None of above

✓ Q16) * 1/1

What does the following C-statement declare?

```
int ( * f) (int * ) ;
```

- A function that takes an integer pointer as argument and returns an integer.
- A function that takes an integer as argument and returns an integer pointer.
- A pointer to a function that takes an integer pointer as argument and returns an integer. ✓
- A function that takes an integer pointer as argument and returns a function pointer

✓ Q64) * 1/1

What will be output when you will execute following c code?

```
#include<stdio.h>
void main(){
    short num[3][2]={3,6,9,12,15,18};
    printf("%d  %d",*(num+1)[1],**(num+2));
}
```

- (A) 12 18
- (B) 18 18
- (C) 15 15 ✓
- (D) 12 15
- (E) Compilation error

✓ Q63) * 1/1

```
void main()
{
     struct bitfield
     {
          unsigned a:5;
          unsigned c:5;
          unsigned b:6;

     }bit;
     char *p;
     struct bitfield *ptr,bit1={1,3,3};
     p=&bit1;
     p++;
     clrscr();
     printf("%d",*p);
     getch();
}
```

- ○ 3
- ○ 5
- ◉ 12 ✓
- ○ none of above

✓ Q37) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    printf("%c\n", 7["IndiaBIX"]);
    return 0;
}
```

- [A]. Error: in printf
- [B]. Nothing will print
- [C]. print "X" ✓
- [D]. print "7"

✓ Q12) * 1/1

Consider this C code to swap two integers and these five statements after it:

```
void swap(int *px, int *py)
{
    *px = *px - *py;
    *py = *px + *py;
    *px = *py - *px;
}
```

S1: will generate a compilation error S2: may generate a segmentation fault at runtime depending on the arguments passed S3: correctly implements the swap procedure for all input pointers referring to integers stored in memory locations accessible to the process S4: implements the swap procedure correctly for some but not all valid input pointers S5: may add or subtract integers and pointers.

- ○ S1
- ○ S2 and S3
- ◉ S2 and S4 ✓
- ○ S2 and S5

✓ Q67) * 1/1

What will be output when you will execute following c code?

```
#include<stdio.h>
typedef struct{
    char *name;
    double salary;
}job;
void main(){
    static job a={"TCS",15000.0};
    static job b={"IBM",25000.0};
    static job c={"Google",35000.0};
    int x=5;
    job * arr[3]={&a,&b,&c};
    printf("%s  %f\t",(3,x>>5-4)[*arr]);
}
double myfun(double d){
       d-=1;
       return d;
}
```

- (A) TCS 15000.000000
- (B) IBM 25000.000000
- (C) Google 35000.000000 ✓
- (D) Compilation error
- (E) None of the above

✓ Q68) * 1/1

What will be output if you will compile and execute the following c code?

```
int * call();
void main(){
int *ptr;
ptr=call();
     clrscr();
printf("%d",*ptr);
}
int * call(){
int a=25;
a++;
return &a;
}
```

- A) 25
- (B) 26
- (C) Any address
- (D) Garbage value ✓
- (E) Compiler error

✓ Q3) * 1/1

What is the output of following program?

```
# include <stdio.h>
void fun(int x)
{
    x = 30;
}

int main()
{
  int y = 20;
  fun(y);
  printf("%d", y);
  return 0;
}
```

- ○ 30
- ◉ 20 ✓
- ○ Compiler error
- ○ Runtime error

✓ Q69) * 1/1

```
void start();
void end();
#pragma startup start
#pragma exit end
int static i;
void main(){
   printf("\nmain function: %d",++i);
}
void start(){
   clrscr();
   printf("\nstart function: %d",++i);
}
void end(){
   printf("\nend function: %d",++i);
   getch();
}
```

```
(a)
main function: 2
start function: 1
end function:3
```

○ A

```
(b)
start function: 1
main function: 2
end function:3
```

◉ B ✓

```
(c)
main function: 2
end function:3
start function: 1
```

C

(d) Compiler error

(e) None of these

✓ Q19) * 1/1

Pick the best statement for the following program snippet:

```
#include <stdio.h>

int main()
{
 int var;  /*Suppose address of var is 2000 */

 void *ptr = &var;
 *ptr = 5;
 printf("var=%d and *ptr=%d",var,*ptr);

 return 0;
}
```

It will print “var=5 and *ptr=2000”

It will print “var=5 and *ptr=5”

It will print “var=5 and *ptr=XYZ” where XYZ is some random address

Compile error ✓

✓ Q40) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    int i, a[] = {2, 4, 6, 8, 10};
    change(a, 5);
    for(i=0; i<=4; i++)
        printf("%d, ", a[i]);
    return 0;
}
void change(int *b, int n)
{
    int i;
    for(i=0; i<n; i++)
        *(b+1) = *(b+i)+5;
}
```

- [ ] [A]. 7, 9, 11, 13, 15
- [x] [B]. 2, 15, 6, 8, 10 ✓
- [ ] [C]. 2 4 6 8 10
- [ ] [D]. 3, 1, -1, -3, -5

✓ Q28)What would be the equivalent pointer expression for referring the array element a[i][j][k][l] *1/1

- [ ] A. ((((a+i)+j)+k)+l)
- [x] B. *(*(*(*(a+i)+j)+k)+l) ✓
- [ ] C. (((a+i)+j)+k+l)
- [ ] D. ((a+i)+j+k+l)

✓ Q14) * 1/1

Predict the output of following program

```
#include<stdio.h>
int main()
{
    int a = 12;
    void *ptr = (int *)&a;
    printf("%d", *ptr);
    getchar();
    return 0;
}
```

- ○ 12
- ◉ Compiler Error ✓
- ○ Runt Time Error
- ○ 0

✓ Q46)Are the expression *ptr++ and ++*ptr are same? * 1/1

- ○ True
- ◉ False ✓

✓ Q39) * 1/1

What will be the output of the program ?

```
#include<stdio.h>
power(int**);
int main()
{
    int a=5, *aa; /* Address of 'a' is 1000 */
    aa = &a;
    a = power(&aa);
    printf("%d\n", a);
    return 0;
}
power(int **ptr)
{
    int b;
    b = **ptr***ptr;
    return (b);
}
```

- [A]. 5
- [B]. 25 ✓
- [C]. 125
- [D]. Garbage value

✓ Q49)Is the NULL pointer same as an uninitialised pointer? * 1/1

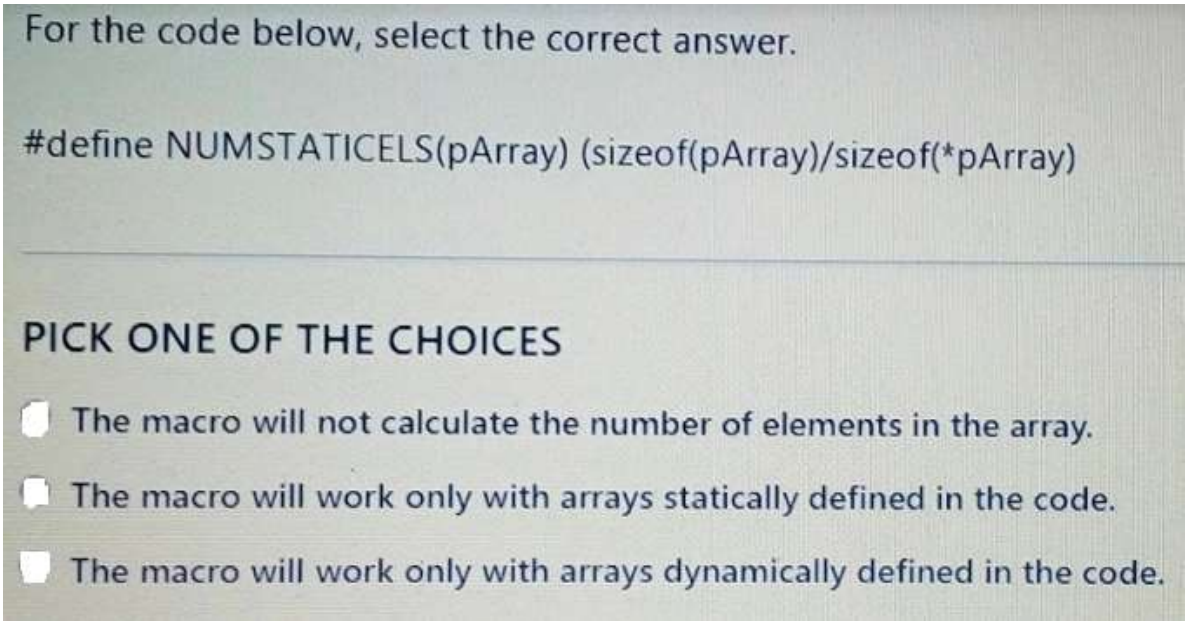
- Yes
- No ✓

✓ Q17) * 1/1

```c
#include <stdio.h>
#define print(x) printf("%d ", x)
int x;
void Q(int z)
{
    z += x;
    print(z);
}
void P(int *y)
{
    int x = *y + 2;
    Q(x);
    *y = x - 1;
    print(x);
}
main(void)
{
    x = 5;
    P(&x);
    print(x);
}
```

- (●) 12 7 6 ✓
- ( ) 22 12 11
- ( ) 14 6 6
- ( ) 7 6 6

✓ Q2) * 1/1


For the code below, select the correct answer.
#define NUMSTATICELS(pArray) (sizeof(pArray)/sizeof(*pArray)
PICK ONE OF THE CHOICES
The macro will not calculate the number of elements in the array.
The macro will work only with arrays statically defined in the code.
The macro will work only with arrays dynamically defined in the code.

- (●) A ✓
- ( ) B
- ( ) C

✓ Q47) * 1/1

The following program reports an error on compilation.

```
#include<stdio.h>
int main()
{
    float i=10, *j;
    void *k;
    k=&i;
    j=k;
    printf("%f\n", *j);
    return 0;
}
```

- ○ True
- ◉ False ✓

✓ Q13) * 1/1

```
int f(int x, int *py, int **ppz)
{
  int y, z;
  **ppz += 1;
   z  = **ppz;
  *py += 2;
   y = *py;
   x += 3;
   return x + y + z;
}

void main()
{
   int c, *b, **a;
   c = 4;
   b = &c;
   a = &b;
   printf("%d ", f(c, b, a));
   return 0;
}
```

- ○ 18
- ◉ 19 ✓
- ○ 21
- ○ 22

✓ Q50) * 1/1

Will the program compile in Turbo C?

```
#include<stdio.h>
int main()
{
    int a=10, *j;
    void *k;
    j=k=&a;
    j++;
    k++;
    printf("%u %u\n", j, k);
    return 0;
}
```

- ○ Yes
- ◉ No ✓

✓ Q21)Consider the size of int as two bytes and size of char as one byte. Predict the output of the following code . Assume that the machine is little-endian. *1/1

Consider the following C code

```
int main()
{
   int a = 300;
   char *b = (char *)&a;
   *++b = 2;
   printf("%d ",a);
   return 0;
}
```

- (●) 556 ✓
- ( ) 300
- ( ) Runtime Error
- ( ) Compile Time Error

✓ Q11) * 1/1

```
#include<stdio.h>
void f(int *p, int *q)
{
  p = q;
  *p = 2;
}
int i = 0, j = 1;
int main()
{
  f(&i, &j);
  printf("%d %d n", i, j);
  getchar();
  return 0;
}
```

- ○ 2 2
- ○ 2 1
- ○ 0 1
- ◉ 0 2 ✓

✓ Q27)Which of the following option is correct? * 1/1

Consider following two C - program :
P1 :

```
int main()
{
    int (*ptr)(int ) = fun;
    (*ptr)(3);
    return 0;
}

int fun(int n)
{
  for(;n > 0; n--)
    printf("GeeksQuiz ");
  return 0;
}
```

P2 :

```
int main()
{
    void demo();
    void (*fun)();
    fun = demo;
    (*fun)();
    fun();
    return 0;
}

void demo()
{
    printf("GeeksQuiz ");
}
```

- ○ P1 printed "GeeksQuiz GeeksQuiz" and P2 printed "GeeksQuiz GeeksQuiz"
- ○ P1 printed "GeeksQuiz GeeksQuiz" and P2 gives compiler error
- ◉ P1 gives compiler error and P2 printed "GeeksQuiz GeeksQuiz" ✓
- ○ None of the above

✓ Q35) * 1/1

What will be the output of the program?

```
#include<stdio.h>

int main()
{
    int arr[3] = {2, 3, 4};
    char *p;
    p = arr;
    p = (char*)((int*)(p));
    printf("%d, ", *p);
    p = (int*)(p+1);
    printf("%d", *p);
    return 0;
}
```

- [A]. 2, 3
- [B]. 2, 0 ✓
- [C]. 2, Garbage value
- [D]. 0, 0

✓ Q18)What's the size returned for each of sizeof() operator? * 1/1

Assume *int* is 4 bytes, *char* is 1 byte and *float* is 4 bytes. Also, assume that pointer size is 4 bytes (i.e. typical case)

```
char *pChar;
int *pInt;
float *pFloat;

sizeof(pChar);
sizeof(pInt);
sizeof(pFloat);
```

- (●) 4 4 4 ✓
- ( ) 1 4 4
- ( ) 1 4 8
- ( ) None of the above

✓ Q60) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
    int * p , b;
    b = sizeof(p);
    printf("%d" , b);
return 0;
}
```

- (A) 2
- (B) 4 ✓
- (C) 8
- (D) Compilation error
- (E) None of above

✓ Q4) * 1/1

Output of following program?

```
#include <stdio.h>

int main()
{
    int *ptr;
    int x;

    ptr = &x;
    *ptr = 0;

    printf(" x = %dn", x);
    printf(" *ptr = %dn", *ptr);

    *ptr += 5;
    printf(" x  = %dn", x);
    printf(" *ptr = %dn", *ptr);

    (*ptr)++;
    printf(" x = %dn", x);
    printf(" *ptr = %dn", *ptr);

    return 0;
}
```

x = 0
*ptr = 0
x = 5
*ptr = 5
x = 6
*ptr = 6

◉ A ✓

x = garbage value
*ptr = 0
x = garbage value
*ptr = 5
x = garbage value
*ptr = 6

○ B

x = 0
*ptr = 0

C x = 5
*ptr = 5
x = garbage value
*ptr = garbage value

x = 0
*ptr = 0

D x = 0
*ptr = 0
x = 0
*ptr = 0

✓ Q36) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    char *str;
    str = "%d\n";
    str++;
    str++;
    printf(str-2, 300);
    return 0;
}
```

- [A]. No output
- [B]. 30
- [C]. 3
- [D]. 300 ✓

✓ Q42) * 1/1

Point out the error in the program

```
#include<stdio.h>

int main()
{
    int a[] = {10, 20, 30, 40, 50};
    int j;
    for(j=0; j<5; j++)
    {
        printf("%d\n", a);
        a++;
    }
    return 0;
}
```

- [A]. Error: Declaration syntax
- [B]. Error: Expression syntax
- [C]. Error: LValue (left hand side) required ✓
- [D]. Error: Rvalue (right hand side) required

✓ Q8)What is the output of above program? * 1/1

```
#include<stdio.h>
int main()
{
    int a;
    char *x;
    x = (char *) &a;
    a = 512;
    x[0] = 1;
    x[1] = 2;
    printf("%dn",a);
    return 0;
}
```

- (●) A) Machine dependent ✓
- ( ) B) 513
- ( ) C) 258
- ( ) D) Compiler Error

✓ Q5) * 1/1

Consider a compiler where int takes 4 bytes, char takes 1 byte and pointer takes 4 bytes.

```
#include <stdio.h>

int main()
{
    int arri[] = {1, 2 ,3};
    int *ptri = arri;

    char arrc[] = {1, 2 ,3};
    char *ptrc = arrc;

    printf("sizeof arri[] = %d ", sizeof(arri));
    printf("sizeof ptri = %d ", sizeof(ptri));

    printf("sizeof arrc[] = %d ", sizeof(arrc));
    printf("sizeof ptrc = %d ", sizeof(ptrc));

    return 0;
}
```

- ○ A) sizeof arri[] = 3 sizeof ptri = 4 sizeof arrc[] = 3 sizeof ptrc = 4
- ○ B) sizeof arri[] = 12 sizeof ptri = 4 sizeof arrc[] = 3 sizeof ptrc = 1
- ○ C) sizeof arri[] = 3 sizeof ptri = 4 sizeof arrc[] = 3 sizeof ptrc = 1
- ◉ D) sizeof arri[] = 12 sizeof ptri = 4 sizeof arrc[] = 3 sizeof ptrc = 4 ✓

✓ Q22)The following 'C' statement : int * f [ ] ( ); declares: * 1/1

- ○ A function returning a pointer to an array of integers.
- ◉ Array of functions returning pointers to integers. ✓
- ○ A function returning an array of pointers to integers.
- ○ An illegal statement.

✓ Q32) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

void fun(void *p);
int i;

int main()
{
    void *vptr;
    vptr = &i;
    fun(vptr);
    return 0;
}
void fun(void *p)
{
    int **q;
    q = (int**)&p;
    printf("%d\n", **q);
}
```

- A. Error: cannot convert from void** to int**
- B. Garbage value
- C. 0 ✓
- D. No output

✓ Q55) * 1/1

What will be output of following program?

```
#include<stdio.h>
int main(){
     int a = 10;
     void *p = &a;
     int *ptr = p;
     printf("%u",*ptr);
     return 0;
}
```

- (A) 10 ✓
- (B) Address
- (C) 2
- (D) Compilation error
- (E) None of above

✓ Q31) * 1/1

What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    int x=30, *y, *z;
    y=&x; /* Assume address of x is 500 and integer is 4 byte size */
    z=y;
    *y++=*z++;
    x++;
    printf("x=%d, y=%d, z=%d\n", x, y, z);
    return 0;
}
```

- A. x=31, y=502, z=502
- B. x=31, y=500, z=500
- C. x=31, y=498, z=498
- D. x=31, y=504, z=504 ✓

✓ Q10) * 1/1

```c
#include<stdio.h>
void fun(int arr[])
{
  int i;
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  for (i = 0; i < arr_size; i++)
      printf("%d ", arr[i]);
}

int main()
{
  int i;
  int arr[4] = {10, 20 ,30, 40};
  fun(arr);
  return 0;
}
```

- ○ 10 20 30 40
- ◉ Machine Dependent ✓
- ○ 10 20
- ○ Nothing