Question 1 of 10

```
#include <stdio.h>
int main(void)
{
        int num=10;
        int *ptr=&num;
        ++*ptr++;
        printf("%d",num);
        printf("%d",*--ptr);
}
```

- Ivalue required error
- 10 garbage
- 11 12
- 11 11

Question 2 of 10

What will be the o/p of the following code

```
#include <stdio.h>
int num=10;
void update(int *ptr)
{
        *ptr+=num;
}
int main(void)
{
        int num=100;
        printf("%d\n",num);
        update(&num);
        printf("%d\n",num);
}
```

- 100 100
- 100 10
- 100 110
- 100 garbage

Question 3 of 10

What will be the o/p of the following code

```
#include <stdio.h>
int* update(int *ptr)
{
        int number=10;
        number=number + *ptr;
        return &number;
}
int main(void)
{
        int *ptr;
        int num=10;
        ptr = update(&num);
        printf("%d",*ptr);

}
```

- 10
- 20
- Compiletime Error
- Value at ptr will try to access an address which is already deallocated , So It will be garbage (can be 0).

Question 4 of 10

```
const int *ptr;
Which is true about above statement
I ptr is constant pointer pointing to constant integer value
II ptr is non constant pointer pointing to constant integer value
III ptr is constant pointer
IV ptr is non constant pointer pointing to non constant integer value
```

- II and III
- Only I
- All the statments
- Only II

Question 5 of 10

What will be the o/p of the following code

```
#include <stdio.h>
int main(void)
{
        int arr[5]={10,20,30,40,50};
        int *ptr;
        ptr = arr+2;
        *ptr=33;
        *ptr++;
        printf("%d",--*ptr);
}
```

- ○ 31
- ○ 20
- ○ 39
- ○ 40

What will be the o/p of the following code

```
#include <stdio.h>
int main(void)
{

        int num=10;
        int *ptr=&num;;
        int **pptr=NULL;

        pptr=&ptr+1;

        printf("%d",**--pptr+1);
        return 0;
}
```

- ○ 10
- ○ Garbage
- ○ Runtime Error
- ○ 11

Question 7 of 10

What will be the o/p of the following code
If P is a pointer to an integer and T is a pointer to a character then scale factor of P will be

- same as that of scale factor of T
- greater than that of scale factor of T
- less than that of scale factor of T
- None of the above

Question 8 of 10

```
#include <stdio.h>
int main(void)
{
        char num=256;
        int *ptr=&num;
        *ptr++;
        int *ptr2=--ptr;
        printf("%d",*(char *)ptr2);
}
```

- Garbage
- 10
- 0
- 11

Question 9 of 10

What will be the output of following code snippet?

```
#include <stdio.h>
 void display(int*);
 int main()
 {
     int i = 10, *p = &i;
     display(p++);
      printf("%d\n", *--p);
  }
void display(int *p)
{
  *p == 11;
   printf("%d\n", *p);
  }
```

- 10 6684232
- 10 11
- 11 11
- 10 10

Question 10 of 10

What will be the Output?

```
#include <stdio.h>
int main()            //assume x address is 2000 and ptr address  is 2004
{
    int x = 0 ;
    int *ptr = &x;
    *ptr += 5;
    printf("\n x  = %d", x);
    printf(" *ptr = %d", *ptr);
    (*ptr)++;
    printf("\n x  = %d", x);
    printf(" *ptr = %d", *ptr++);
    printf("\n Difference= %d", ptr - &x);
    return 0;
}
```

- ○ x=5 *ptr=5 x=6 *ptr =6 Difference=1
- ○ x=5 *ptr=garbage x=6 *ptr =garbage Difference=2
- ○ x=5 *ptr=garbage x=6 *ptr =6 Difference=-4
- ○ x=5 *ptr=5 x=6 *ptr =6 Difference=4

1.
```
#include <stdio.h>
int main(void)
{
        int num=10;
        int *ptr=&num;
        ++*ptr++;
        printf("%d",num);
        printf("%d",*--ptr);
}
```

Answers

1. lvalue required error

2. 10 garbage

3. 11 12

4. 11 11

2. What will be the o/p of the following code

```
#include <stdio.h>
int num=10;
void update(int *ptr)
{
        *ptr+=num;
}
int main(void)
{
        int num=100;
        printf("%d\n",num);
        update(&num);
        printf("%d\n",num);
}
```

Answers

1. 100 100

2. 100 10

3. 100 110

4. 100 garbage

3. What will be the o/p of the following code

```
#include <stdio.h>
int* update(int *ptr)
{
        int number=10;
        number=number + *ptr;
        return &number;
}
int main(void)
{
        int *ptr;
        int num=10;
        ptr = update(&num);
        printf("%d",*ptr);

}
```

Answers

1. 10

2. 20

3. Compiletime Error

4. Value at ptr will try to access an address which is already deallocated , So It will be garbage (can be 0).

4. const int *ptr;

Which is true about above statement

I ptr is constant pointer pointing to constant integer value

II ptr is non constant pointer pointing to constant integer value

III ptr is constant pointer

IV ptr is non constant pointer pointing to non constant integer value

Answers

1. II and III

2. Only I

3. All the statments

4. Only II

5. What will be the o/p of the following code

```
#include <stdio.h>
int main(void)
{
        int arr[5]={10,20,30,40,50};
        int *ptr;
        ptr = arr+2;
        *ptr=33;
        *ptr++;
        printf("%d",--*ptr);
}
```

Answers

1. 31

2. 20

3. 39

4. 40

6. What will be the o/p of the following code

```
#include <stdio.h>
int main(void)
{

        int num=10;
        int *ptr=&num;;
        int **pptr=NULL;

        pptr=&ptr+1;

        printf("%d",**--pptr+1);
        return 0;
}
```

Answers

1. 10

2. Garbage

3. Runtime Error

4. 11

7. What will be the o/p of the following code

If P is a pointer to an integer and T is a pointer to a character then scale factor of P will be

Answers

1. same as that of scale factor of T

2. greater than that of scale factor of T

3. less than that of scale factor of T

4. None of the above

8. #include <stdio.h>

```
int main(void)
{
        char num=256;
        int *ptr=&num;
        *ptr++;
        int *ptr2=--ptr;
        printf("%d",*(char *)ptr2);
}
```

Answers

1. Garbage

2. 10

3. 0

4. 11

9. What will be the output of following code snippet?

```
#include <stdio.h>
 void display(int*);
 int main()
 {
     int i = 10, *p = &i;
     display(p++);
      printf("%d\n", *--p);
  }
void display(int *p)
{
  *p == 11;
   printf("%d\n", *p);
  }
```

Answers

1. 10 6684232

2. 10 11

3. 11 11

4. 10 10

10. What will be the Output?

```
#include <stdio.h>
int main()           //assume x address is 2000 and ptr address  is 2004
{
    int x = 0 ;
    int *ptr = &x;
    *ptr += 5;
    printf("\n x  = %d", x);
    printf(" *ptr = %d", *ptr);
    (*ptr)++;
    printf("\n x  = %d", x);
    printf(" *ptr = %d", *ptr++);
    printf("\n Difference= %d", ptr - &x);
    return 0;
}
```

Answers

1. x=5 *ptr=5 x=6 *ptr =6 Difference=1

2. x=5 *ptr=garbage x=6 *ptr =garbage Difference=2

3. x=5 *ptr=garbage x=6 *ptr =6 Difference=-4

4. x=5 *ptr=5 x=6 *ptr =6 Difference=4