C.S Prilims 2019

Answer key.

1. Both types of arguments and order of arguments are different.
2. First In Last Out
3. Seekg()
4. X.X'=0
5. TELNET
6. Abstract Class: It is a class which can act as base class only during inheritance and not used for creating objects.
7.

Array	Linked List
Memory allocated are sequential	Memory allocated may not be sequential.
Element can be randomly accessed	Element has to be sequentially accessed
Memory size is fixed	Memory size is dynamic
Insertion and deletion of element is slow relatively as shifting is required	Insertion and deletion of element is easier faster and efficient

8. xyz*+pq*r+s*+
9. Webpage: A **webpage** is a document commonly written in HTML (Hypertext Markup Language) that is accessible through the Internet or other networks using an Internet browser.
10. **Bridge** is a **networking** device that connects two or more LANs together following same protocol.
11. -2 2 0 1
12. i) They have same name as class name.
    ii) They are defined in public section of class.
    iii) They are called when program control goes out of program, function or a block.
    iv) They don't have return type
13. Sand Rock Hill ~Hill ~Rock ~Sand
14.
```
void pattern(int n)
{       int i,j;
        for(i=1;i<=n;i++)
        {
                for(j=1;j<=i;j++)
                cout<<i*j<<" ";
                cout<<endl;

        }

}
```

15. Stack is a data structure in which elements are inserted and deleted at the same end.
    Queue is a data structure in which elements are inserted at one end known as rear and elements are deleted at another end know as front.

16. Statement 1: p.seekp(ios::end);
    Statement 2: p.tellp();
17. #include<iostream.h>
    #include<fstream.h>
    using namespace std;
    int main()

    {
     ifstream a("story.txt");
     char c;
     int count=0,v=0,x=0;

     while(a.get(c))
     if(isalpha(c))
          if(c!='a'|| c!='e'|| c!='i'|| c!='c'|| c!='u') v++; else x++;
    cout<<v<<x;
    }
18. XNOR GATE:

A	B	F
0	0	1
0	1	0
1	0	0
1	1	1

SOP : F=A'B'+AB

POS: F=(A+B')(A'+B)

19. (A+B)'= (A'.B')
    Let X=A+B
    According to complementary law X + X'=1
    L.H.S: (A+B)+(A+B)'
          (A+B)+ A'B'
          A+B+A' (Redundancy Law)
          1+B (Identity Law)
          1=R.H.S (dominance law)

20. FTP:File Transfer Protocol is a part of TCP/IP and enables files to be transferred between computers.
    The Basics Steps to use FTP are
    a) Connect to FTP server
    b) Navigate the file structure to find the file the user wants.
    c) Transfer the file.
    Files on FTP Server are compressed which enables more files to be stored or the server and make transfer time shorter

21. Low Cost, Waste Reduction, Speed, Ease of use, Record Maintenance

22.
```
#include <iostream>
#include<math.h>
using namespace std;
int main ( )
{
        int n,c=1;
        cout<<"Enter n :";
        cin>>n;

        for(int i=2;i<=sqrt(n);i++)
        if(n%i==0) {c=0; break;}
        if(c)cout<<"\nprime"; else cout<<"\ncomposite";
        return 0;
}
```

OR

```
#include <iostream>
using namespace std;

int main ( )
{
int n,sum=1;
cout<<"Enter n :";
cin>>n;
sum+=n;
for(int i=2;i<=n/2;i++)
if(n%i==0) sum+=i;
cout<<endl<<sum;
return 0;
}
```

23.
```
distance(distance &s, distance &t)
{
        x=s.x+t.x+(s.x+t.x) /12;
        y=(s.y+t.y) % 12;
}
```

24.

Assumption:

I) Let P indicate the postfix expression to be evaluated

ii) Let STACK indicate the stack used to evaluate .

Iii) Let # indicate bottom of stack

iv) Let) indicate the end of postfix expression.

v) Let VALUE indicate the final value of postfix expression

Symbol Scanned	Operation Performed	Stack Content
5	PUSH(5)	5#
4	PUSH(4)	4,5#
6	PUSH(6)	6,4,5#
+	A=POP(6),B=POP(4) PUSH(10)	10,5#
*	A=POP(10),B=POP(5) PUSH(50)	50#
4	PUSH(4)	4,50#
9	PUSH(9)	9,4,50#
3	PUSH(3)	3,9,4,50#
/	A=POP(3),B=POP(9) PUSH(3)	3,4,50#
+	A=POP(3),B=POP(4) PUSH(7)	7,50#
*	A=POP(7),B=(50) PUSH(350)	350#
)	POP(350)	VALUE=350

25.

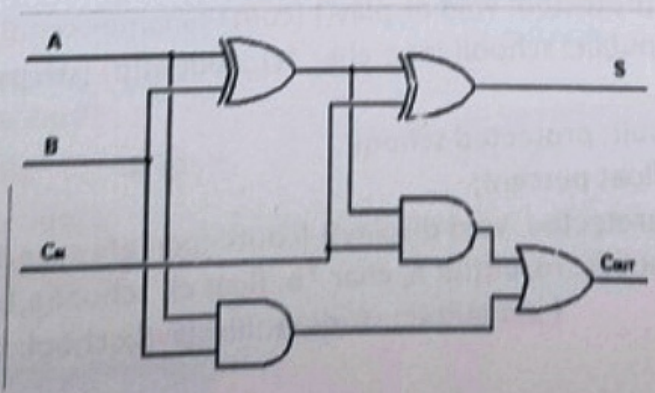
```
void task()
{
        fstream a;
        a.open("address.dat",ios::binary|ios::in);
        student x;
        while(a.read((char*)&x,sizeof(x))
        if(!x.check("panaji"))  x.putdata();
        a.close();
}
```

26.

	INPUTS		OUTPUTS	
A	B	CIN	COUT	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$S = A'B'C + A'BC' + AB'C' + ABC$
$= C(A'B' + AB) + C'(A'B + AB')$
$= C(A \oplus B)' + C'(A \oplus B)$
$= A \oplus B \oplus C$

$C_{OUT} = A'BC + AB'C + ABC' + ABC$
$= C(A'B + AB') + AB(C + C')$
$= C(A \oplus B) + AB$



27.
```
#include <iostream>
#include <string.h>
using namespace std;
class student
{       int roll;
        protected: void display() {cout<<roll<<endl;}
        public: student(int r) { roll=r; display();}
};

class school
{       char name[30];
        protected: void display() {cout<<name<<endl;}
        public: school( char *s) {strcpy(name,s);display();}
};

class result
{       float percent;
        student x;
        school y;
        public: result(int a, char *b, float c): x(a),y(b)
                { percent=c;}
                void display() {cout<<percent<<endl;}
};
int main()
{ result z(1000,"xyx",78.95); z.display(); return 0;}
```

OR

27.
```
#include <iostream>
#include <string.h>
using namespace std;
class student
{       int roll;
        protected: void display() {cout<<roll<<endl;}
        public: student(int r) { roll=r; }
};

class school: protected student
{       char sname[30];
        protected: void display() {cout<<sname<<endl;}
        public: school( int p,char *s):student(p) {strcpy(sname,s);}
};
class result: protected school
{       float percent;
        protected: void display() {cout<<percent<<endl;}
        public: result(int a, char *b, float c): school(a,b)
                { percent=c; student::display();school::display();display();}
};

int main()
{ result z(1000,"xyx",78.95); return 0;}
```

28.
```
void list::insertion()
{       node *ptr=start,*temp;
        int i,pos;

        temp=new node;
        cout<<"\nEnter the value in new node to be inserted : ";
        cin>> temp->data;
        temp->next=NULL;
        cout<<"\nEnter the position number where the node has to inserted :";
        cin>>pos;

        if(pos==1)
        {       temp->next=start;
                start=temp;
        }
        else
        {       for(i=1;i<=pos-2;i++)
                ptr=ptr->next;

                temp->next=ptr->next;
                ptr->next=temp;
          }
}
```

OR

```
void list::creation()
{ int i,n;
node *ptr,*temp;
cout<<"\n Enter the total no. of nodes in the linked list:";
cin>>n;

for(i=1;i<=n;i++)
{       temp= new node;
        cout<<"Enter the value: ";
        cin>>temp->data;
        temp->next=NULL;

        if(i==1) {start=ptr=temp;}
        else
        {       ptr->next=temp;
                ptr=temp;
        }}}
```