Q4/ Given a circlular queue of size (10) contains (10) elements,
write coplete program to convert the even values to any array of size(10).

```
#include <iostream>

using namespace std;

int const size = 10;
int j = 0;

void insertion(int cq[size], int &f, int &r)
{
      int value;
      int i;

      for(i = 0; i <= 9; i++)
      {
            if(r == size - 1 && f == 0)
                  cout<<"C Queue full !! insertion is not possible\n";

            else if(f - r == 1)
                  cout<<"C Queue full !! insertion is not possible\n";

            else if(r == size - 1 && f > 0)
            {
                  cout<<"Enter any value : ";
                  cin>>value;

                  r = 0;
                  cq[r] = value;
            }

            else
            {
                  cout<<"Enter any value : ";
                  cin>>value;

                  ++r;
                  cq[r] = value;
            }

            if(f == -1)
                  f = 0;
      }
}


int con_Cq_arr(int cq[size], int arr_even[size], int &f, int &r)
{
      int value, i;

      if(f == -1)
            cout<<"C Queue is empty\n";

      else if(f == r)
            {
                  if(cq[i] % 2 == 0)
```

```
                {
                        value = cq[r];
                        f = -1;
                        r = -1;

                        arr_even[j] = value;
                        ++j;
                }
        }

    else if(r > f)
    {
        for(i = f; i <= r; i++)
        {
                if(cq[i] % 2 == 0)
                {
                        value = cq[i];

                        arr_even[j] = value;
                        ++j;
                }

        }
    }

    else
    {
        for(i = f; i <= size - 1; i++)
        {
                if(cq[i] % 2 == 0)
                {
                        value = cq[i];

                        arr_even[j] = value;
                        ++j;
                }
        }
        for(i = 0; i <= r; i++)
        {
                if(cq[i] % 2 == 0)
                {
                        value = cq[i];

                        arr_even[j] = value;
                        ++j;
                }
        }
    }

    for(i = 0; i < j; i++)
        cout<<arr_even[i]<<"\t";
    cout<<"\n";
}


void print(int cq[size], int &f, int &r)
{
    int i;
```

```
	if(f == -1)
		cout<<"C Queue is empty\n";

	else if(f == r)
		cout<<cq[r]<<"\n";

	else if(r > f)
	{
		for(i = f; i <= r; i++)
		{
			cout<<cq[i]<<"\t";
		}
		cout<<"\n";
	}

	else
	{
		for(i = f; i <= size - 1; i++)
		{
			cout<<cq[i]<<"\t";
		}
		for(i = 0; i <= r; i++)
		{
			cout<<cq[i]<<"\t";
		}
		cout<<"\n";
	}

}


int main()
{
	int cq[size];
	int arr_even[size];
	int r = -1;
	int f = -1;

	insertion(cq, f, r);
	cout<<endl;
	print(cq, f, r);
	cout<<endl;
	con_Cq_arr(cq, arr_even, f, r);
	cout<<endl;
	print(cq, f, r);


    return 0;
}
```

Q3:) Having a circular Queue of size (10) , F=5 ,R=2 ,write program segment with draw to:
a) convert all values of circular queue to empty queue of size (7)?
b) Delete (3) elements from new queue?
c) Print the final state of Queue?

---

a/

```
#include <iostream>

using namespace std;

int const size1 = 10;
int const size2 = 7;

void con_C_Q(int cq1[size1], int &f1, int &r1, int cq2[size2], int &f2, int &r2)
{
	int i;
	int value;

	for(i = 0; i <= r1; i++)
	{
		++r2;
		cq2[r2] = cq1[i];
	}

	if(f2 == -1)
		f2 = 0;

	for(i = f1; i <= size1 - 1; i++)
	{
		if(r2 == 6&& f2 == 0)
		{
			value = cq2[f2];
			++f2;

			r2 = 0;
			cq2[r2] = cq1[i];

		}
		else
		{
			++r2;
			cq2[r2] = cq1[i];
		}
	}
}

void printQ2(int cq2[size2], int f2, int r2)
{
	int i;

	for(i = f2; i <= size2 - 1; i++)
		cout<<cq2[i]<<"\t";

	for(i = 0; i == r2; i++)
		cout<<cq2[i]<<"\t";
```

```
      cout<<"\n";
}

int main()
{
      int cq1[size1] = {1, 2, 3, NULL, NULL, 6, 7, 8, 9, 10};
      int cq2[size2];

      int f1 = 5;
      int r1 = 2;
      int f2 = -1;
      int r2 = -1;

      cout<<"\n";
      con_C_Q(cq1, f1, r1, cq2, f2, r2);
      cout<<"\n";
      printQ2(cq2, f2, r2);
      cout<<"\n";

    return 0;
}
```

Q/Q1: Given a circular queue of size (8) contain (4) elements, Do the following (using program segment) :
A) add (5) elements? Then
B) Convert the negative values to empty stack of size (5)? Then
C) print the final state of the stack?

```
#include <iostream>

using namespace std;

int const size1 = 8;
int const size2 = 5;

void insertio(int cq[size1], int &f, int &r)
{
      int value;
      int i;

      for(i =0; i <= 4; i++)
      {
            if(r == 7 && f == 0)
            {
                  value =cq[f];
                  ++f;

                  cout<<"Enter any number : ";
                  cin>>value;

                  r = 0;
                  cq[r] = value;
            }
            else
            {
                  cout<<"Enter any number : ";
                  cin>>value;

                  ++r;
                  cq[r] = value;
            }
      }
}

void co_s(int cq[size1], int stack[size2], int &top, int &f, int &r)
{
      int i;
      int value;

      for(i = f; i <= 7; i++)
      {
            if(cq[i] < 0)
            {
                  if(top == 4)
                  {
                        value = stack[top];
                        --top;
                        ++top;
```

```
                        stack[top] = cq[i];
                  }
                  else
                  {
                        ++top;
                        stack[top] = cq[i];
                  }
            }
      }
      for(i = 0; i == r; i++)
      {
            if(cq[i] < 0)
            {
                  if(top == 4)
                  {
                        value = stack[top];
                        --top;
                        ++top;
                        stack[top] = cq[i];
                  }
                  else
                  {
                        ++top;
                        stack[top] = cq[i];
                  }
            }
      }
}

void printQ(int cq[size1], int f, int r)
{
      int i;

      for(i = f; i <= 7; i++)
      {
            cout<<cq[i]<<"\t";
      }
      for(i = 0; i == r; i++)
      {
            cout<<cq[i]<<"\t";
      }
}

void prints(int stack[size2], int top)
{
      int i;

      for(i = 0; i <= top; i++)
      {
            cout<<stack[i]<<"\t";
      }
}


int main()
{
      int cq[size1] = {1, 2, -3, -4};
      int stack[size2];
```

```
      int f = 0;
      int r = 3;
      int top = -1;

      insertio(cq, f, r);
      cout<<"\n";

      co_s(cq, stack, top, f, r);
      cout<<"\n";

      printQ(cq, f, r);
      cout<<"\n";

      prints(stack, top);
      cout<<"\n";

    return 0;
}
```

Q1/ Given a circular queue of size (10) contain (10) elements, Convert the positive values to empty stack of size (10), and the negative values to empty Queue of size (10) , final print the stack and the queue ?

```
#include <iostream>

using namespace std;

int const size = 10;


void insertio(int cq[size], int &f1, int &r1)
{
      int value;
      int i;

      for(i =0; i <= 9; i++)
      {
            cout<<"Enter any number : ";
            cin>>value;

                  ++r1;
                  cq[r1] = value;
      }

      if(f1 == -1)
            f1 = 0;
}

void co_s(int cq[size], int stack[size], int &top, int &f1, int &r1, int
queue[size], int &f2, int &r2)
{
      int i;
      int value;

      for(i = f1; i <= r1; i++)
      {
            if(cq[i] >= 0)
            {
                  ++top;
                  stack[top] = cq[i];
            }

            else
            {
                  ++r2;
                  queue[r2] = cq[i];

                  if(f2 == -1)
                        f2 = 0;
            }
      }

}

void printcQ(int cq[size], int f1, int r1)
{
```

```
	int i;

	for(i = f1; i <= r1; i++)
	{
		cout<<cq[i]<<"\t";
	}
}

void prints(int stack[size], int top)
{
	int i;

	for(i = 0; i <= top; i++)
	{
		cout<<stack[i]<<"\t";
	}
}

void printq(int queue[size], int f2, int r2)
{
	int i;

	for(i = f2; i <= r2; i++)
	{
		cout<<queue[i]<<"\t";
	}
}


int main()
{
	int cq[size];
	int stack[size];
	int queue[size];

	int f1 = -1;
	int r1 = -1;
	int f2 = -1;
	int r2 = -1;
	int top = -1;

	insertio(cq, f1, r1);
	cout<<"\n";

	co_s(cq, stack, top, f1, r1, queue, f2, r2);
	cout<<"\nC Queue :\n";

	printcQ(cq, f1, r1);
	cout<<"\nStack :\n";

	prints(stack, top);
	cout<<"\nQueue :\n";

	printq(queue, f2, r2);
	cout<<"\n";

    return 0;
}
```

```
#include <iostream>

using namespace std;

int const size = 5;


void insertion(int cq[size], int &f, int &r)
{
      int value;
      int i;

      for(i = 0; i <= 4; i++)
      {
            if(r == size - 1 && f == 0)
                  cout<<"C Queue full !! insertion is not possible\n";

            else if(f - r == 1)
                  cout<<"C Queue full !! insertion is not possible\n";

            else if(r == size - 1 && f > 0)
            {
                  cout<<"Enter any value : ";
                  cin>>value;

                  r = 0;
                  cq[r] = value;
            }

            else
            {
                  cout<<"Enter any value : ";
                  cin>>value;

                  ++r;
                  cq[r] = value;
            }

            if(f == -1)
                  f = 0;
      }
}

void deletion(int cq[size], int &f, int &r)
{
      int value, i;

      for(i = 0; i <= 4; i++)
      {
            if(f == -1)
            cout<<"C Queue is empty !! is deletion is not possible \n";

            else if(f == r)
            {
                  value = cq[f];
                  f = -1;
                  r = -1;
            }
```

```
            else if(f == size - 1 && f > r)
            {
                  value = cq[f];
                  f = 0;
            }

            else
            {
                  value = cq[f];
                  ++f;
            }
            }

}

void print(int cq[size], int &f, int &r)
{
      int i;

      if(f == -1)
            cout<<"C Queue is empty\n";

      else if(f == r)
            cout<<cq[r]<<"\n";

      else if(r > f)
      {
            for(i = f; i <= r; i++)
            {
                  cout<<cq[i]<<"\t";
            }
            cout<<"\n";
      }

      else
      {
            for(i = f; i <= size - 1; i++)
            {
                  cout<<cq[i]<<"\t";
            }
            for(i = 0; i <= r; i++)
            {
                  cout<<cq[i]<<"\t";
            }
            cout<<"\n";
      }

}


int main()
{
      int cq[size];
      int r = -1;
      int f = -1;

      insertion(cq, f, r);
      cout<<endl;
      print(cq, f, r);
```

```
      cout<<endl;
      deletion(cq, f, r);
      print(cq, f, r);


    return 0;
}
```

```
#include <iostream>

using namespace std;

int const size = 6;

void insertion(int cq[size], int &f, int &r)
{
	int value;

	if(r == size - 1 && f == 0)
		cout<<"C Queue full !! insertion is not possible\n";
	else if(f - r == 1)
		cout<<"C Queue full !! insertion is not possible\n";
	else if(r == size - 1 && f > 0)
	{
		cout<<"Enter any value : ";
		cin>>value;

		r = 0;
		cq[r] = value;
	}
	else
	{
		cout<<"Enter any value : ";
		cin>>value;

		++r;
		cq[r] = value;
	}

	if(f == -1)
		f = 0;
}

void deletion(int cq[size], int &f, int &r)
{
	int value;

	if(f == -1)
		cout<<"C Queue is empty !! is deletion is not possible \n";

	else if(f == r)
	{
		value = cq[f];
		f = -1;
		r = -1;
	}

	else if(f == size - 1 && f > r)
	{
		value = cq[f];
		f = 0;
	}

	else
	{
		value = cq[f];
		++f;
```

```
	}

}

void print(int cq[size], int &f, int &r)
{
	int i;

	if(f == -1)
		cout<<"C Queue is empty\n";

	else if(f == r)
		cout<<cq[r]<<"\n";

	else if(r > f)
	{
		for(i = f; i <= r; i++)
		{
			cout<<cq[i]<<"\t";
		}
		cout<<"\n";
	}

	else
	{
		for(i = f; i <= size - 1; i++)
		{
			cout<<cq[i]<<"\t";
		}
		for(i = 0; i <= r; i++)
		{
			cout<<cq[i]<<"\t";
		}
		cout<<"\n";
	}

}


int main()
{
	int cq[size];
	int r = -1;
	int f = -1;
	int x;

	for( ; x != 4; )
	{
		cout<<"1- insert \n";
		cout<<"2- delete \n";
		cout<<"3- print \n";
		cout<<"4- exit \n";

		cout<<"Enter your choose :";
		cin>>x;

		switch(x)
		{
			case 1: insertion(cq, f, r); break;
```

```
                case 2: deletion(cq, f, r); break;
                case 3: print(cq, f, r); break;
                default: cout<<"Error\n";
            }
      }

    return 0;
}
```