Inventory

- An inventory system using C++.

Submitted by	**USN**
Moris Takhellambam	1BM20EC173
Hidangmayum Rahul Dev Sharma	1BM20EC054
Owais Javed	1BM20EC095

item.h

```
#pragma once            //this particular header file to be included only once when included

#include <iostream>
using namespace std;

class Item
{
  protected:
    string name;      //every item has a name
    int size, price;  //every item has size and price

  public:
  int getSize(){        //getter function of item size
    return size;
  }
  string getName(){     //getter function of item name
    return name;
  }
};

class Gun : public Item
{
  protected:
    float damage, fireRate;
};

class Ordinance : public Item
{
  protected:
    float throwDistance, damage, range;
};

class Heal : public Item
{
  protected:
    float healHP;
};
```

rifle.h

```
#pragma once
#include "item.h"


class Rifle : public Gun
{
   protected:
      float weight;


   public:
   Rifle()
   {
      name = "Rifle";
      size = 16;
      price = 3000;
      damage = 40;
      fireRate = 10;
      weight = 30;
   }
};
```

pistol.h

```
#pragma once
#include "item.h"

class Pistol : public Gun
{
  protected:
    float weight;

  public:
  Pistol()
  {
    name = "Pistol";
    size = 4;
    price = 500;
    damage = 10;
    fireRate = 1;
    weight = 10;
  }
};
```

grenade.h

```
#pragma once
#include "item.h"

class Grenade : public Ordinance
{
   public:
   Grenade()
   {
      name = "Grenade";
      size = 2;
      price = 200;
      throwDistance = 80;
      damage = 50;
      range = 20;
   }
};
```

molly.h

```
#pragma once
#include "item.h"


class Molly : public Ordinance
{
   public:
   Molly()
   {
      name = "Molly";
      size = 2;
      price = 150;
      throwDistance = 60;
      damage = 30;
      range = 30;
   }
};
```

bandage.h

```
#pragma once
#include "item.h"

class Bandage : public Heal
{
   public:
   Bandage()
   {
      name = "Bandage";
      size = 1;
      price = 100;
      healHP = 20;
   }
};
```

plates.h

```
#pragma once
#include "item.h"

class Plates : public Heal
{
   public:
   Plates()
   {
      name = "Plates";
      size = 1;
      price = 100;
      healHP = 40;
   }
};
```

allitems.h

```
/* a file that has all the item files included so that it could be included where all items are required
at once */

#pragma once


#include "bandage.h"

#include "grenade.h"

#include "molly.h"

#include "pistol.h"

#include "plates.h"

#include "rifle.h"
```

inventory.h

```
#pragma once
#include "allitems.h"

#include <iostream>
using namespace std;

class Inventory
{
   int size;                      //maintaining size of the inventory so that it has limit on the no. of items
                                  //that can be carried

   Bandage band1;      //instanciating items
   Grenade gred1;
   Molly moll1;
   Pistol pist1;
   Plates plat1;
   Rifle rifl1;

   int bandCount = 0;     //initialising a variable that keep count of each item as 0
   int gredCount = 0;
   int mollCount = 0;
   int pistCount = 0;
   int platCount = 0;
   int rifCount = 0;

   public:

   void addItem(){
      int choice;
```

```
    cout << "Choose the item to be added: " << endl
        << "\t1.Rifle\n\t2.Pistol\n\t3.Grenade\n\t4.Molly\n\t5.Bandage\n\t6.Plates" << endl;
    cin >> choice;

    switch (choice)
    {
        case 1:{
            rifCount += 1;

            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";        /*returns full if the current size of the items times the
quantity is more than the size of the inventory*/
                rifCount -= 1;
                break;
            }
            else break;
        }
        case 2:{
            pistCount += 1;

            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";
                pistCount -= 1;
                break;
            }
            else break;
        }
        case 3:{
            gredCount += 1;
            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";
                gredCount -= 1;
                break;
            }
            else break;
        }
        case 4:{
            mollCount += 1;

            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";
                mollCount -= 1;
                break;
            }
            else break;
        }
        case 5:{
            bandCount += 1;

            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";
                bandCount -= 1;
                break;
            }
            else break;
        }
        case 6:{
            platCount += 1;


            if(rifCount*rifl1.getSize() + pistCount*pist1.getSize() + gredCount*gred1.getSize() +
    mollCount*moll1.getSize() + bandCount*band1.getSize() + platCount*plat1.getSize() > size) {
                cout << "Inventory is full.\n";
                platCount -= 1;
                break;
            }
            else break;
        }


        default:    cout << "Invalid entry!";
            break;
    }
}


void deleteItems(){
    int choice;
    cout << "What would you like to remove: " << endl
        << "\t1.Rifle\n\t2.Pistol\n\t3.Grenade\n\t4.Molly\n\t5.Bandage\n\t6.Plates" << endl;
    cin >> choice;                              //read the item no. to be deleted to choice


    switch (choice)
    {
        case 1: {
            if(rifCount == 0)
                    cout << "There is no rifle left.\n";        //says invalid if the quantity is already zero
            else rifCount -= 1;                                 //reduce the quantity by one


            break;
        }
```

```
case 2: {
   if(pistCount == 0)
            cout << "There is no pistol left.\n";
   else pistCount -= 1;

   break;
}

case 3: {
   if(gredCount == 0)
            cout << "There is no grenade left.\n";
   else gredCount -= 1;

   break;
}

case 4: {
   if(mollCount == 0)
            cout << "There is no molly left.\n";
   else mollCount -= 1;

   break;
}

case 5: {
   if(bandCount == 0)
            cout << "There is no bandage left.\n";
   else bandCount -= 1;

   break;
```

```
        }


        case 6: {
          if(platCount == 0)
                  cout << "There are no plates left.\n";
          else platCount -= 1;


          break;
        }


        default:{
          cout << "Invalid Entry.";
          break;
        }
    }
}

// void displayItems(){                            //without using iostream
//    cout<< endl << "Item\t\t\t\tQuantity\n----\t\t\t\t--------\n"
//       << rifl1.getName() << "\t\t\t\t" << rifCount << endl
//       << pist1.getName() << "\t\t\t\t" << pistCount << endl
//       << gred1.getName() << "\t\t\t\t" << gredCount << endl
//       << moll1.getName() << "\t\t\t\t" << mollCount << endl
//       << band1.getName() << "\t\t\t\t" << bandCount << endl
//       << plat1.getName() << "\t\t\t\t" << platCount << endl << endl;
// }

void displayItems(){                             //using iostream
    cout.setf(ios::left, ios::adjustfield);
    cout.width(10);
    cout << "\nItem";
```

```
cout.setf(ios::right, ios::adjustfield);
cout.width(30);
cout << "Quantity\n";

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "1. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << rifl1.getName();
cout.setf(ios::left, ios::adjustfield);
cout.width(4);
cout << rifCount << endl;

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "2. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << pist1.getName();
cout.setf(ios::left, ios::adjustfield);
cout.width(4);
cout << pistCount << endl;

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "3. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << gred1.getName();
```

```
cout.setf(ios::left, ios::adjustfield);
cout.width(4);
cout << gredCount << endl;

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "4. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << moll1.getName();
cout.setf(ios::left, ios::adjustfield);
cout.width(4);
cout << mollCount << endl;

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "5. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << band1.getName();
cout.setf(ios::left, ios::adjustfield);
cout.width(4);
cout << bandCount << endl;

cout.setf(ios::right, ios::adjustfield);
cout.width(4);
cout << "6. ";
cout.setf(ios::left, ios::adjustfield);
cout.width(29);
cout << plat1.getName();
cout.setf(ios::left, ios::adjustfield);
```

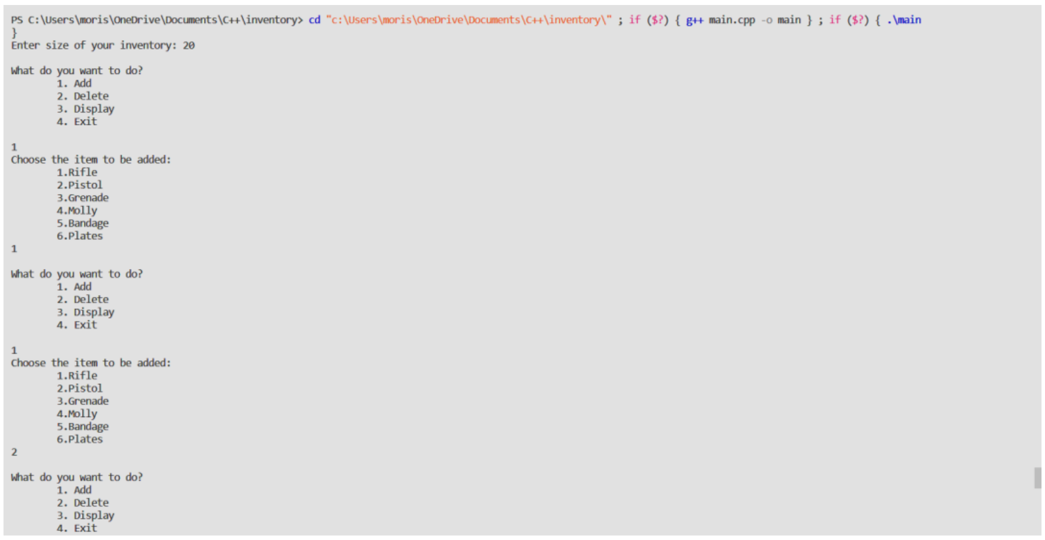
```
        cout.width(4);
        cout << platCount << endl;


    }


    Inventory(){
        size = 30;


            while(1){
            int choice;
            cout <<"\nWhat do you want to do?\n\t1. Add\n\t2. Delete\n\t3. Display\n\t4. Exit" << endl
<< endl;
            cin >> choice;


            switch (choice)
            {
            case 1: {
                addItem();
            }
                break;


            case 2: {
                deleteItems();
            }


            case 3: {
                displayItems();
            }


            case 4: {
                break;
            }

            default:
                break;                       // if invalid entry is made, it breaks and runs the while block again
            }

            if (choice == 4) break;       // the invalid entry is checked for 4, if 4 then the while loop breaks
            }
    }

    Inventory(int invSize){
                                /* overloading the constructor function if the user wants to enter the size of
                        the inventory manually */
        size = invSize;

            while(1){
            int choice;
            cout << "\nWhat do you want to do?\n\t1. Add\n\t2. Delete\n\t3. Display\n\t4. Exit" << endl
<< endl;
            cin >> choice;

            switch (choice)
            {
            case 1: {
                addItem();
                break;
            }

            case 2: {
                deleteItems();
                break;
            }
```

```
            case 3: {
              displayItems();
              break;
            }

            case 4: {
              break;
            }

            default:
              break;                  // if invalid entry is made, it breaks and runs the while block again
            }

            if (choice == 4) break;   // the invalid entry is checked for 4, if 4 then the while loop breaks
            }
        }
    };
```

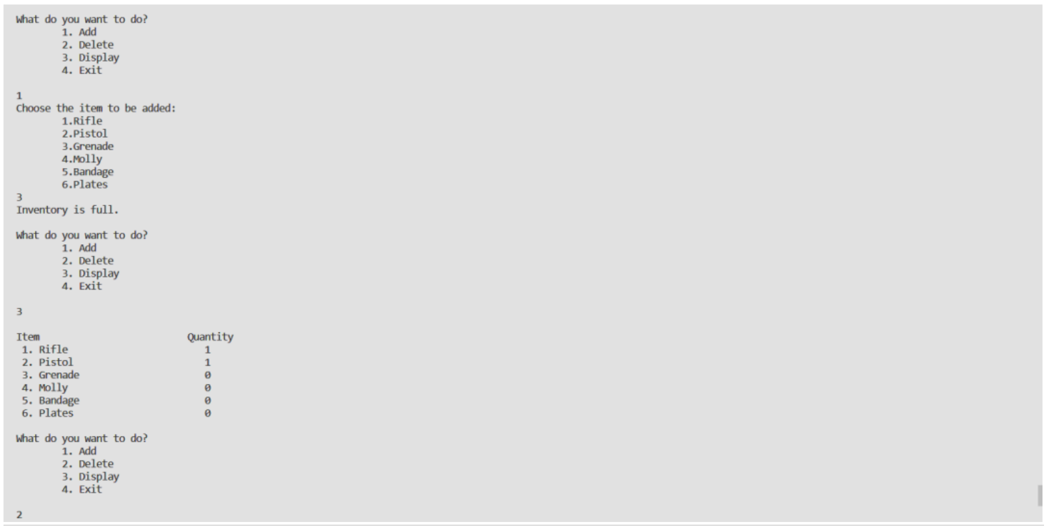
main.cpp

```
#include "inventory.h"


int main(){


    int userSize;

    cout << "Enter size of your inventory: ";

    cin >> userSize;

     Inventory invUser(userSize);




   // cout << "Default inventory: \n";

    //Inventory invDef;




    return 0;

}
```

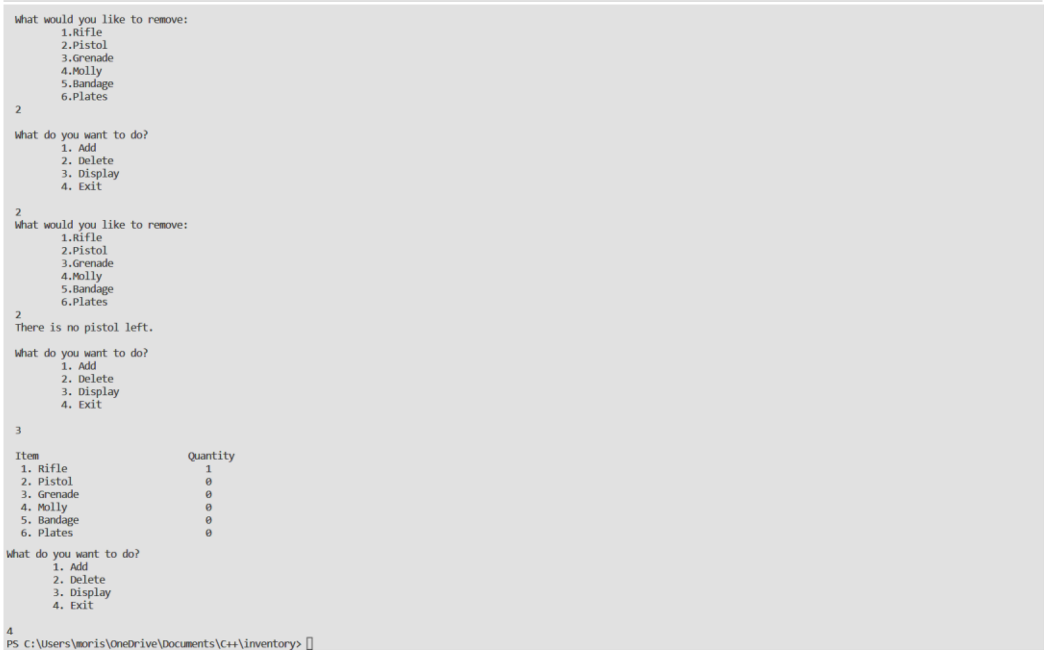
Output



```
PS C:\Users\moris\OneDrive\Documents\C++\inventory> cd "c:\Users\moris\OneDrive\Documents\C++\inventory\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main
}
Enter size of your inventory: 20

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

1
Choose the item to be added:
        1.Rifle
        2.Pistol
        3.Grenade
        4.Molly
        5.Bandage
        6.Plates
1

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

1
Choose the item to be added:
        1.Rifle
        2.Pistol
        3.Grenade
        4.Molly
        5.Bandage
        6.Plates
2

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit
```



```
What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

1
Choose the item to be added:
        1.Rifle
        2.Pistol
        3.Grenade
        4.Molly
        5.Bandage
        6.Plates
3
Inventory is full.

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

3

Item                          Quantity
 1. Rifle                        1
 2. Pistol                       1
 3. Grenade                      0
 4. Molly                        0
 5. Bandage                      0
 6. Plates                       0

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

2
```



```
What would you like to remove:
        1.Rifle
        2.Pistol
        3.Grenade
        4.Molly
        5.Bandage
        6.Plates
2

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

2
What would you like to remove:
        1.Rifle
        2.Pistol
        3.Grenade
        4.Molly
        5.Bandage
        6.Plates
2
There is no pistol left.

What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

3

Item                          Quantity
 1. Rifle                        1
 2. Pistol                       0
 3. Grenade                      0
 4. Molly                        0
 5. Bandage                      0
 6. Plates                       0
What do you want to do?
        1. Add
        2. Delete
        3. Display
        4. Exit

4
PS C:\Users\moris\OneDrive\Documents\C++\inventory> 
```