

Vector Using Bloodshed Dev C++

By Janine Bouyssounouse

Bloodshed Dev C++ is a free program to make writing and compiling C++ programs easy to do. It makes executable files quickly so programs can be shared with others as soon as they are written and debugged. Bloodshed Dev C++ can be downloaded from the website:

<http://www.bloodshed.net/devcpp.html>.

There is more than one way to start writing programs in Dev C++. This document discusses using a project for each program. This encourages using an organized file structure to keep each program in a separate location for future use. Remember that reusing code is a good idea and it is nice to be able to find the folder with everything for a program in one place.

Click on the File menu and select New – Project. Choose Console Application for this program. Type in a name for the project, such as Display Input, in the name field in the bottom left corner of the window. Click on the OK button. Choose a place on the computer to save the project. It's a good idea to make a folder for each project. Once the project is saved, some basic items show up on the screen. These items give a shell of a C++ program and include things that are unique to Bloodshed Dev C++ as well.

```
#include <iostream>    - This tells the compiler to include files.
#include <stdlib.h>    - This tells the compiler to include files.

using namespace std;  - This tells the compiler that you will be using key
                      words that are included in namespace std.

int main(int argc, char *argv[])    - This starts the main function.
{                                     - Main is contained between { and }.

    system("PAUSE");                - This leaves the console window open until
                                     you are ready to close it. Otherwise you
                                     wouldn't be able to see the results of the
                                     program. Not all compilers require this.

    return 0;                       - This is how the main program ends.
}                                     - This signifies the end of the code for main.
```

The code for the main part of the program will be typed between the first curly brace ({} and the system("PAUSE") line of code. The input and print functions for this program will be outside of this area, but the main program will refer to them.

Programs have a need to store and access data. There are many different ways to handle this. Larger programs have a need for the use of databases. But smaller programs can get away with other methods. This document is an example of using the vector ability in C++. I find it easier to use than an array. I find it more flexible.

This program will store information and then access that information to display it to the user in a random format as many times as the user wants to see the information. A random writing prompt program is used as the example. A predetermined list of writing prompts are entered into the vector, then the user is given the opportunity to see as many of them randomly selected as they want. A do while loop with a yes/no question function is used to make this happen. This is an example of a function calling a function that doesn't even appear in the main function at all.

Type in the first lines of the program by clicking at the beginning of the first line and pressing enter a couple of times, then moving to the first blank line. Type in these first lines of code:

```
// This program uses a vector to store data to be used in the program
// then randomly displays that data until the user wants to stop seeing it.
```

The next line of code can be a comment stating your name as the programmer and the date the program was written.

```
// Written by Janine Bouyssounouse on 12/22/08
```

Type the following line of code after the using namespace statement:

```
void welcome();
```

A welcome statement will be displayed to explain what this program does. There is nothing sent to or received from the welcome function, so the void and empty parentheses are used in the function prototype.

On the last line of the program, we will comment the line to show that the main function is finished, so that it is not confused with the other functions listed after it.

The last line of code should look like this:

```
} // end main
```

Next we will start typing the welcome function at the end of the program, outside of the curly braces for the main function.

Skip a line and type:

```
// welcome function displays an opening message to  
// explain the program to the user  
void welcome()
```

Notice the comments are listed on the lines before the start of the function. The first line of the function looks exactly like the function prototype, except for the missing semicolon at the end.

On the next line of code, type in the function:

```
{  
  cout << "Welcome to the Random Prompt Generator.\n\n";  
  cout << "Take a set amount of time to write about the\n";  
  cout << "listed prompt. Do this daily and your writing\n";  
  cout << "will improve. Keep a journal of your writings\n";  
  cout << "to see your improvement over time as well as a\n";  
  cout << "way to get ideas for future writings.\n\n";  
} // end of welcome function
```

This function uses no variables at all.

Call the welcome function in the main function to display the information to the user.

To do this, type the following code after the first curly brace in the main function and before the `system("PAUSE")` line of code:

```
welcome(); // This calls the welcome function
```

Vectors, strings and random numbers will be used in this program. We will also be seeding the random function with the time from the computer, so we need to add more include statements at the top of the program. This tells the program where to find the preset functions that are already part of the C++ language. It saves time to only load the information as needed by programs. Type these lines of code after `#include <stdlib.h>` before the main function:

```
#include <string>
#include <vector>
#include <cstdlib>
#include <ctime>
```

Type the following line of code after the `void welcome();` statement in the function prototype section before the main function:

```
vector<string> loadPrompts();
```

`vector<string>` means there is a vector made up of string values returned to the main function. `loadPrompts` is the name of the function. The empty parentheses show that nothing is being passed to the `loadPrompts` function from the main program. The semicolon shows the end of the line of code for the function prototype.

Think of the vector as a box where things are stored, like a shoe box with a whole lot of slips of paper in it. Each piece of paper has one of the strings or in this case a writing prompt on it. It is similar to the variable concept of a box with only one value in it. So a vector is like having a lot of things in one place. This means it has its own set of rules to follow because it is a lot harder to handle a whole box of slips of paper than it is to handle just one single piece of information.

At the end of the program, type in the `loadPrompts` function:

```
// loadPrompts creates and loads a vector with writing prompts
```

```

// a vector is returned to where the function was called
vector<string> loadPrompts()
{
    vector<string> prompts; // declares vector of strings called prompts

    // adds strings to the vector prompts with the use of the push_back feature
    prompts.push_back("Convince someone you should be president.");
    prompts.push_back("Describe a trip you took with your family.");
    prompts.push_back("Make up a game and list the rules.");
    prompts.push_back("Write a letter to your favorite star.");
    prompts.push_back("Write a letter to your best friend.");
    prompts.push_back("My favorite color is...");
    prompts.push_back("I feel like going to...");

    return prompts; // sends back the prompts vector
} // end loadPrompts function

```

The first line in the function declares the vector as a string vector with the name of prompts. Other types of vectors can be declared as well, just follow this method for declaring them.

Then the function continues by adding data to the vector. The `push_back` feature is one of the special features that can be used with vectors. It is a useful way to load a vector when you don't know how big it will be. It is more flexible than an array.

The list of writing prompts has been shortened here and the final listing of the program has a much longer list of them. Feel free to add whatever writing prompts you would like to use for yourself or others to customize the application for your own use.

Since the vector is being sent back to where it was called, a vector needs to be declared in the main function to receive the vector from this function. Type the following line of code in the main function before the call to the `welcome` function:

```
vector<string> writingPrompts; // declares a vector of strings
```

Now call the loadPrompts function in the main function. Type the following code after the welcome function is called in the main function:

```
writingPrompts = loadPrompts(); // calls loadPrompts and receives vector
```

Another function needs to be created, but it won't be called from the main function. It is the askYesNo function which will determine when it is time to stop showing random writing prompts and will be called from the function that displays the writing prompts which will be written after this function.

Type the following line of code at the end of the list of function declarations before the main function:

```
char askYesNo(string question);
```

This function returns a character (char) which will be the y or n answer to the question that is sent to the function (string question).

Now type the function at the very bottom of the program:

```
// askYesNo function takes any question it is sent and continues
// to ask it until it gets a y or n answer, then the answer
// is returned to where the function was called
char askYesNo(string question)
{
    char response; // variable is declared to receive input from user

    // loops until either a y or and n is entered by the user
    do
    {
        cout << question << " (y/n): "; // writes question and y/n prompt
        cin >> response;
        cout << endl;
    } while (response != 'y' && response != 'n'); // condition to end loop

    return response; // returns the y or n back to where function was called
} // end askYesNo function
```

The do while loop takes care of making sure the response is in the correct format of either a y or n so that the function receiving the information can properly use it. The way the function is set up, any string can be used when calling the function. The string just has to be worded in the form of a yes or no question, including the question mark. The string needs to be in quotation marks when it is sent to the function.

Now the function is ready to be called from the next function we will create. Type the following line of code at the end of the list of function prototypes before the main function:

```
void displayPrompt(vector<string> writingPrompts);
```

Nothing is returned from this function, so it is a void function. A string vector will be sent to the function to allow it to do its job with the list of writing prompts.

Type in the function at the end of the program:

```
// displayPrompt function is passed the vector and then randomly displays
one
// prompt at a time until the user says no to wanting to see another prompt
void displayPrompt(vector<string> writingPrompts)
{
    srand(time(0)); // seeds the random function

    // main program loop continues until user does not say y to repeat question
    do
    {
        int randNumber = rand();

        // gets a random number based on how many prompts are in the vector
        int number = (randNumber % writingPrompts.size()) + 1;

        cout << "Prompt:\n"; // displays an intro to the prompt

        // displays prompt based on random number determined above
        cout << writingPrompts[number] << endl;
```

```
// calls askYesNo function with the question to be asked and continues
// looping as long as the answer is y
} while (askYesNo("\n\nWould you like to see another prompt?") == 'y');
} // end displayPrompt function
```

This function calls a function from within a function as a condition of the do while loop. The askYesNo function could have also been called using a string variable with the question loaded into the string. This is a matter of choice by the programmer. I chose to eliminate one variable, but it might be easier to use another variable and pass it to the displayPrompts function when it is called to make the question easier to find in the code when it needs to be changed.

Now the function needs to be called from the main function. Type the following line of code after writingPrompts = loadPrompts(); in the main function:

```
displayPrompt(writingPrompts); // sends vector to displayPrompt function
```

The program is finished. Here is the code:

```
// This program uses a vector to store data to be used in the program
// then randomly displays that data until the user wants to stop seeing it.
// Written by Janine Bouyssounouse on 12/22/08

#include <iostream>
#include <stdlib.h>
#include <string>
#include <vector>
#include <cstdlib>
#include <ctime>

using namespace std;

void welcome();
vector<string> loadPrompts();
char askYesNo(string question);
void displayPrompt(vector<string> writingPrompts);
```

```

int main(int argc, char *argv[])
{
    vector<string> writingPrompts; // declares a vector of strings

    welcome(); // This calls the welcome function
    writingPrompts = loadPrompts(); // calls loadPrompts and receives vector
    displayPrompt(writingPrompts); // sends vector to displayPrompt function

    system("PAUSE");
    return 0;
} // end main function

// welcome function displays an opening message to
// explain the program to the user
void welcome()
{
    cout << "Welcome to the Random Prompt Generator.\n\n";
    cout << "Take a set amount of time to write about the\n";
    cout << "listed prompt. Do this daily and your writing\n";
    cout << "will improve. Keep a journal of your writings\n";
    cout << "to see your improvement over time as well as a\n";
    cout << "way to get ideas for future writings.\n\n";
} // end of welcome function

// loadPrompts creates and loads a vector with writing prompts
// a vector is returned to where the function was called
vector<string> loadPrompts()
{
    vector<string> prompts; // declares vector of strings called prompts

    // adds strings to the vector prompts with the use of the push_back feature
    prompts.push_back("Convince someone you should be president.");
    prompts.push_back("Describe a trip you took with your family or
friends.");
    prompts.push_back("Make up a game and list the rules.");
    prompts.push_back("Write a letter to your favorite star.");
    prompts.push_back("Write a letter to your best friend.");
    prompts.push_back("Pick a side and defend it... Is it better to have a dog or
a cat as a pet?");
}

```

```
prompts.push_back("My favorite color is...");
prompts.push_back("I feel like going to...");
prompts.push_back("I like my hair today because...");
prompts.push_back("Tell a story about... What's behind door number
two?");
prompts.push_back("Write about something you like to do in your spare
time.");
prompts.push_back("Write about a walk you took recently.");
prompts.push_back("I do/don't like to go swimming.");
prompts.push_back("Describe your favorite shoes.");
prompts.push_back("School is...");
prompts.push_back("Use the words book, truck, and string in a story.");
prompts.push_back("Describe your first memory.");
prompts.push_back("Describe your favorite place to go.");
prompts.push_back("Describe how you relax.");
prompts.push_back("Describe the hardest thing you have ever done.");
prompts.push_back("Describe your favorite teacher.");
prompts.push_back("I like/don't like math because...");
prompts.push_back("I like/don't like English because...");
prompts.push_back("Convince someone your favorite subject in school is
the best subject.");
prompts.push_back("Write a story using the words yellow, house, and
fish.");
prompts.push_back("If you lived on the moon, what would it be like.");
prompts.push_back("If you could go anywhere in the world, where would
you go and why.");
prompts.push_back("What would it be like for a dolphin to kiss you?");
prompts.push_back("Do you like having people read out loud to you?
Why?");
prompts.push_back("Take a walk on a path. What is it like and why are
you there?");
prompts.push_back("Explain why you would or would not want to learn
how to play an instrument.");
prompts.push_back("Explain the steps to clean your home.");
prompts.push_back("Describe the career would you like to have when you
are an adult.");
prompts.push_back("My favorite magazine...");
prompts.push_back("Use greed, laughter, and ice cream in a story.");
prompts.push_back("Draw a picture of your dream car and then describe it
in detail.");
```

```
prompts.push_back("Explain how you helped someone.");
prompts.push_back("Write a story starting with: I overheard you
saying...");
prompts.push_back("Laughter is the best medicine.");
prompts.push_back("Write a thank you note.");
prompts.push_back("Write a poem about your favorite relative.");
prompts.push_back("Write a review for your favorite movie.");
prompts.push_back("Write a story about a cafe you own.");
prompts.push_back("Write a story including the words cheap, bubble, and
umbrella.");
prompts.push_back("What does it mean to be free?");
prompts.push_back("Compare and contrast a pool table and a coffee
table.");
prompts.push_back("Defend your right to choose your own career.");
prompts.push_back("Write a poem starting with: Today is...");
prompts.push_back("The internet is...");
prompts.push_back("Take a stand. Do aliens exist? Yes or no.");
prompts.push_back("Describe what you had for breakfast. Could it have
been better?");
prompts.push_back("What would you like to learn if time and money were
no object?");
prompts.push_back("You're at your 10 year reunion. What is it like?");
prompts.push_back("Old McDonald had a farm...");
prompts.push_back("And the dish ran away with the spoon...");
prompts.push_back("Write a plan for the next five years of your life.");
prompts.push_back("Write a story using the words green, fan, and
doorway.");
prompts.push_back("How does it feel to do something nice for
someone?");
prompts.push_back("Write a poem about a body of water.");
prompts.push_back("What would your life be like without electricity?");
prompts.push_back("Write the pros and cons of war.");
prompts.push_back("Look, it's a bird...");
prompts.push_back("What would it be like if everyone lived forever?");
prompts.push_back("If the birds and the bees talked to each other, what
would they say?");
prompts.push_back("Use the words building, book, and blueberry in a
story.");
prompts.push_back("Use the words purple, peanut, and plunder in a
story.");
```

```
prompts.push_back("Use the words Friday, friend, and feast in a story.");
prompts.push_back("Why should people learn more than one language?");
prompts.push_back("What would happen if no one believed what you
said?");
prompts.push_back("Why is farming important?");
prompts.push_back("What does it mean to talk with your hands?");
prompts.push_back("Describe your body language. How does it help you
communicate?");
prompts.push_back("What would you do to change your life?");
prompts.push_back("What would your life be like on another planet?");
prompts.push_back("Describe a time when you were happy.");
prompts.push_back("Describe a time when you were angry.");
prompts.push_back("If you were a movie star, what would your life be
like?");
prompts.push_back("The grass is always greener...");
prompts.push_back("The sky is falling...");
prompts.push_back("How would life be different if you lived on the
moon?");
prompts.push_back("Compare and contrast living in a tree and living
underground.");
prompts.push_back("Should people have pets? Yes or no?");
prompts.push_back("Is school important? Why or why not?");
prompts.push_back("Why do kids need babysitters?");
prompts.push_back("If at first you don't succeed...");
prompts.push_back("Are you feisty? Why or why not?");
prompts.push_back("What are the steps to doing the laundry?");
prompts.push_back("Describe the steps to make your favorite food.");
prompts.push_back("Many people used to believe the world was flat.
Why?");
prompts.push_back("Compare and contrast making music to listening to
music.");
prompts.push_back("What is a weekend warrior?");
prompts.push_back("Are computers helpful? Why or why not?");
prompts.push_back("You've just burned dinner, what happens next?");
prompts.push_back("You just found out your friend lied to you. What do
you do?");
prompts.push_back("You're giving a speech. Are you excited or afraid?
Why?");
prompts.push_back("You just fell down the stairs and no one is around...");
prompts.push_back("You're in a jungle. What do you see?");
```

```

prompts.push_back("You're lost. What do you do?");

return prompts; // sends back the prompts vector
} // end loadPrompts function

// askYesNo function takes any question it is sent and continues
// to ask it until it gets a y or n answer, then the answer
// is returned to where the function was called
char askYesNo(string question)
{
    char response; // variable is declared to receive input from user

    // loops until either a y or and n is entered by the user
    do
    {
        cout << question << " (y/n): "; // writes question and y/n prompt
        cin >> response;
        cout << endl;
    } while (response != 'y' && response != 'n'); // condition to end loop

    return response; // returns the y or n back to where function was called
} // end askYesNo function

// displayPrompt function is passed the vector and then randomly displays
// one
// prompt at a time until the user says no to wanting to see another prompt
void displayPrompt(vector<string> writingPrompts)
{
    srand(time(0)); // seeds the random function

    // main program loop continues until user does not say y to repeat question
    do
    {
        int randNumber = rand();

        // gets a random number based on how many prompts are in the vector
        int number = (randNumber % writingPrompts.size()) + 1;

        cout << "Prompt:\n"; // displays an intro to the prompt
    }
}

```

```
// displays prompt based on random number determined above
cout << writingPrompts[number] << endl;

// calls askYesNo function with the question to be asked and continues
// looping as long as the answer is y
} while (askYesNo("\n\nWould you like to see another prompt?") == 'y');
} // end displayPrompt function
```

Save, compile and run the program to see if it works. Choose Compile and Run from the Execute menu.

Here is a display of the program:

```
Welcome to the Random Prompt Generator.

Take a set amount of time to write about the
listed prompt. Do this daily and your writing
will improve. Keep a journal of your writings
to see your improvement over time as well as a
way to get ideas for future writings.

Prompt:
School is...

Would you like to see another prompt? (y/n): y

Prompt:
If you lived on the moon, what would it be like.

Would you like to see another prompt? (y/n): m

Would you like to see another prompt? (y/n): n

Press any key to continue . . .
```

Please note the askYesNo function was tested with entering the letter m instead of the letter n. The function just looped right back around and asked the question again. The computer doesn't care how many times you mistype something. Computers can be very patient that way...

Now write a program of your own.

Exercise 1: Write a program to print random numbers between one and twenty, but make sure the numbers are written out in words, instead of displaying the numbers from the random number generator. Continue displaying the word numbers until the user wants to stop. Use the `askYesNo` function discussed in this packet. Choose names for the functions that tell what they do.

Sample Code for Exercise 1:

```
// This program uses a vector to store data to be used in the program
// then randomly displays that data until the user wants to stop seeing it.
// Written by Janine Bouyssounouse on 12/22/08

#include <iostream>
#include <stdlib.h>
#include <string>
#include <vector>
#include <cstdlib>
#include <ctime>

using namespace std;

void welcome();
vector<string> loadNumbers();
char askYesNo(string question);
void displayNumbers(vector<string> wordNumbers);

int main(int argc, char *argv[])
{
    vector<string> wordNumbers; // declares a vector of strings

    welcome(); // This calls the welcome function
    wordNumbers = loadNumbers(); // calls loadPrompts and receives vector
    displayNumbers(wordNumbers); // sends vector to displayPrompt function

    system("PAUSE");
    return 0;
} // end main function

// welcome function displays an opening message to
// explain the program to the user
void welcome()
{
    cout << "Random numbers are displayed spelled out";
    cout << " until the user wants it to stop.\n\n";
} // end of welcome function
```

```

// loadNumbers creates and loads a vector with numbers
// a vector is returned to where the function was called
vector<string> loadNumbers()
{
    vector<string> numbers; // declares vector of strings called numbers

    // adds strings to the vector numbers with the use of the push_back feature
    numbers.push_back("one");
    numbers.push_back("two");
    numbers.push_back("three");
    numbers.push_back("four");
    numbers.push_back("five");
    numbers.push_back("six");
    numbers.push_back("seven");
    numbers.push_back("eight");
    numbers.push_back("nine");
    numbers.push_back("ten");
    numbers.push_back("eleven");
    numbers.push_back("twelve");
    numbers.push_back("thirteen");
    numbers.push_back("fourteen");
    numbers.push_back("fifteen");
    numbers.push_back("sixteen");
    numbers.push_back("seventeen");
    numbers.push_back("eighteen");
    numbers.push_back("nineteen");
    numbers.push_back("twenty");

    return numbers; // sends back the numbers vector
} // end loadNumbers function

// askYesNo function takes any question it is sent and continues
// to ask it until it gets a y or n answer, then the answer
// is returned to where the function was called
char askYesNo(string question)
{
    char response; // variable is declared to receive input from user

    // loops until either a y or and n is entered by the user
    do

```

```

{
    cout << question << " (y/n): "; // writes question and y/n prompt
    cin >> response;
    cout << endl;
} while (response != 'y' && response != 'n'); // condition to end loop

return response; // returns the y or n back to where function was called
} // end askYesNo function

// displayNumbers function is passed the vector and then randomly displays
one
// number at a time until the user says no to wanting to see another number
void displayNumbers(vector<string> wordNumbers)
{
    srand(time(0)); // seeds the random function

    // main program loop continues until user does not say y to repeat question
    do
    {
        int randNumber = rand();

        // gets a random number based on how many prompts are in the vector
        int number = (randNumber % wordNumbers.size()) + 1;

        cout << "Random word number:\n"; // displays an intro to the number

        // displays number based on random number determined above
        cout << wordNumbers[number] << endl;

        // calls askYesNo function with the question to be asked and continues
        // looping as long as the answer is y
    } while (askYesNo("\n\nWould you like to see another number?") == 'y');
} // end displayNumbers function

```

Display from Sample Code for Exercise 1:

Random numbers are displayed spelled out until the user wants it to stop.

Random word number:

three

Would you like to see another number? (y/n): y

Random word number:

eleven

Would you like to see another number? (y/n): y

Random word number:

ten

Would you like to see another number? (y/n): y

Random word number:

twelve

Would you like to see another number? (y/n): n

Press any key to continue . . .