Although(хотя) a modern computer is capable(способен) of accepting(принимать) many types of input, it can only operate on digitally formatted data, just as original computers did. Hence(следовательно) software must be created to interface the computer with the various types of input data. Because a computer runs on electricity, data must be stored as a series of on and off patterns.

Computer circuits(схемы) can be in only one or two states: either on (represented by 1)( представлено 1) or off (represented by 0)( представлено 0). Each numeric value is known as a binary digit (bit) and unique combinations of those two bits, are what binary code, or machine language, is called. Different patterns in binary code could then be used to represent various input characters. Once data has been converted to binary form, computers would then apply a software program (applications or apps) to the digital input data, sequentially *(последовательно)* execute the instructions, and successfully process it into information.

Writing software programs in numerical (or digital) format was an immense *(огромной)* task for anyone. Therefore, it was necessary to develop software into a more user-friendly format. As programmers developed existing code, new computer capabilities were noticed and the demand for even more advanced software increased. It inspired programmers to develop more software. Thousands of new programs were being written as swiftly as possible and yet the demand continued to increase. Software types typically fall into 2 categories: system software and application software.

System software controls various internal computer activities. Any *(любое)* software that controls such activities will fall into one of three categories: programming language, operating system and utility software.

*Programming languages* are the various methods of writing computer instructions. The instructions adhere *(придерживаются)* to a particular *(особое)* set of protocols for each language. Through the years, more than 200 languages have been developed, some of which are quite specialized. Some of the most popular languages include BASIC, COBOL, Pascal, C, C++, Visual Basic. But regardless *(невзирая на то)* of which language a program was written in, a computer can only process binary code. Therefore, each language must eventually be converted back to binary code before any instructions can be followed.

High-level languages were developed for two reasons: one – so programmers could work on different computers without having to learn a new assembly language each time, and software written on one computer could be used by another. A compiler (program translator) was used to help solve these problems by translating a program into machine language and checking the program for syntax errors.

Until 1970, IBM bundled *(поставлял в комплекте)* its software with its computers, selling the hardware along *(продавая оборудование вместе с)* with the software needed to run it. IBM began charging a separate fee *(начали взимать дополнительную плату)* for its software, thus opening a market for independent software developers to write programs that would run on IBM machines. By the time the first personal computer (PC), called the Altair hit the market *(вышел на рынок)* in the 1975, there were many well-developed computer languages and competent *(грамотные)* programmers available *(подходящих)* to write software for the new industry.

*Operating systems* have become larger and more sophisticated in response to the capabilities *(в ответ на возможности)* of new hardware and other software. CP/M developed by Gary Kildall in 1973 for Intel Corporation, was the first OS that would run on PCs made by different manufacturers *(изделий)*, and it had the largest number of programs for data and word processing and calculations. Although it was a powerful operating system, few software developers supported it, referring to write for the growing DOS-based market.

*Utility* software expands the performance *(производительность)* of the operating system by adding functions that are not part of the original OS. Utilities perform troubleshooting jobs, inspecting diskettes for damage, file conversion, defragmenting, data compression and file spooling *(буферизация файла)*. Some utility programs, such as Symantec’s Norton Utilities, even retrieve *(извлекают)* data from damaged disks. Utilities can also be used to customize the OS environment.

**Пересказ**

Although a modern computer is capable of accepting many types of input, it can only operate on digitally formatted data, just as original computers did. Hence software must be created to interface the computer with input data. Because a computer runs on electricity, data must be stored as a series of on and off patterns.

Computer circuits can be in only one or two states: either on (represented by 1) or off (represented by 0).

Writing software programs in numerical (or digital) format was an immense task for anyone. Therefore, it was necessary to develop software into a more user-friendly format

System software controls various internal computer activities.

*Programming languages* are the various methods of writing computer instructions. The instructions adhere to a particular set of protocols for each language.

High-level languages were developed for two reasons: one – so programmers could work on different computers without having to learn a new assembly language each time, and software written on one computer could be used by another.

*Operating systems* have become larger and more sophisticated in response to the capabilities of new hardware and other software.

*Utility* software expands the performance of the operating system by adding functions that are not part of the original OS. Utilities perform troubleshooting jobs, inspecting diskettes for damage, file conversion, defragmenting, data compression and file spooling.